Digital Planar Surface Segmentation Using Local Geometric Patterns

Yukiko Kenmochi[1], Lilian Buzer[1], Akihiro Sugimoto[1,2], and Ikuko Shimizu[3]

[1] Université Paris-Est, LABINFO-IGM, UMR CNRS 8049, A2SI-ESIEE, France
[2] National Institute of Informatics, Japan
[3] Tokyo University of Agriculture and Technology, Japan

Abstract. This paper presents a hybrid two-step method for segmenting a 3D grid-point cloud into planar surfaces by using discrete-geometry results. Digital planes contain a finite number of local geometric patterns (LGPs). Such a LGP possesses a set of normal vectors. By using LGP properties, we first reject non-linear points from a point cloud (edge-based step), and then classify non-rejected points whose LGPs have common normal vectors into a planar-surface-point set (region-based step).

1 Introduction

This paper presents a method for segmenting a 3D grid-point cloud into planar surfaces by using discrete-geometry results, providing that the point cloud is obtained from polyhedral objects.

In computer vision, conventional methods for surface segmentation are classified into three categories: region-based, edge-based, and hybrid methods. The first ones merge points having similar region properties calculated from their neighboring points such as normal vectors and curvatures [1]. As calculated properties are sensitive to noise and quantization errors, they cause over-segmentation. The second methods search edges that separate regions by using depth discontinuities [13]. As edges are not always extracted as connected curves, they cause under-segmentation. The third methods are hybrid between the two [10,12]. In particular, for the case where our interesting object is polyhedral, a hybrid method using locally planar points is proposed [10]. In that method, points not locally planar are considered to be potentially edge points.

In discrete geometry, a digital plane is defined as a set of grid points lying between two parallel planes with a small distance [7]. Local geometric patterns (LGPs) appearing on digital planes are called linear LGPs, and their number is finite. It is known that linear LGPs are related to arithmetic planes [8]. Their arithmetic properties were studied for digital plane recognition [4,11], and used to develop region-based methods for digital planar surface segmentation [9]. However, those region-based methods require an incremental plane recognition process, which causes another problem of incremental point tracking.

In this paper, to avoid plane recognition involving incremental point tracking for segmentation, we present a discrete version of the hybrid methods, consisting of an edge-based and a region-based parts, using linear LGPs. Our idea is similar

D. Coeurjolly et al. (Eds.): DGCI 2008, LNCS 4992, pp. 322–333, 2008.

to [10]. We first generate all linear LGPs in a cubic region of $(2k+1) \times (2k+1) \times (2k+1)$ grid points for an arbitrary size k, by using their arithmetic properties. We then reject a point from a grid-point set if its LPG is not linear; as rejected points define candidates of edge points (the edge-based part), the remaining non-rejected points define candidates of planar points. For the region-based part, we use the normal vectors of linear LPGs. Each linear LGP possesses a set of feasible normal vectors, called preimages [3]. We merge non-rejected points whose linear LGPs have common normal vectors to obtain digital planar surfaces. We show our experimental results demonstrating that our method is robust against not only quantization errors but also noise.

2 Digital Plane Patches and Their Preimages

2.1 Digital Planes

Let $\mathbb{R}$ be the set of real numbers. A plane $\mathbf{P}$ in the 3D Euclidean space $\mathbb{R}^3$ is defined by the following expression:

$$\mathbf{P} = \{(x, y, z) \in \mathbb{R}^3 \ : \ \alpha x + \beta y + \gamma z + \delta = 0\} \tag{1}$$

where $\alpha, \beta, \gamma, \delta \in \mathbb{R}$.

Let $\mathbb{Z}$ be the set of integers, so that $\mathbb{Z}^3$ denotes the set of grid points whose coordinates are integers. We now consider a digitization of $\mathbf{P}$, which is a grid-point subset in $\mathbb{Z}^3$, called a digital plane. There are various digitization techniques for $\mathbf{P}$ [7]. In this paper we adopt a grid-line digitization such that

$$\mathbf{D}(\mathbf{P}) = \{(x, y, z) \in \mathbb{Z}^3 \ : \ 0 \leq \alpha x + \beta y + \gamma z + \delta < \omega\} \tag{2}$$

where $\omega = \max(|\alpha|, |\beta|, |\gamma|)$. From the definition, it is obvious that we obtain a unique digital plane $\mathbf{D}(\mathbf{P})$ from any $\mathbf{P}$.

2.2 Preimages of a Digital Plane Patch

Since a point cloud is acquired by a sensor, we can assume that it is bounded. In other words, our grid space $\mathbf{X} \subset \mathbb{Z}^3$ is bounded such that $\mathbf{X} = [X_1, X_2] \times [Y_1, Y_2] \times [Z_1, Z_2]$ where X_i, Y_i, Z_i for $i = 1, 2$ are finite integers. Then, our digital plane $\mathbf{D}(\mathbf{P})$ is also bounded such that

$$\mathbf{D}_{\mathbf{X}}(\mathbf{P}) = \{(x, y, z) \in \mathbf{X} \ : \ 0 \leq \alpha x + \beta y + \gamma z + \delta < \omega\}, \tag{3}$$

called a digital plane patch.

Given $\mathbf{D}_{\mathbf{X}}(\mathbf{P})$, we can find a set of Euclidean planes $\mathbf{P}$ such that the digitization of each $\mathbf{P}$ in $\mathbf{X}$ is equal to $\mathbf{D}_{\mathbf{X}}(\mathbf{P})$. The set of all such $\mathbf{P}$ is called the preimage of $\mathbf{D}_{\mathbf{X}}(\mathbf{P})$ [3]. Note that the correspondence between $\mathbf{D}_{\mathbf{X}}(\mathbf{P})$ and $\mathbf{P}$ is not one-to-one but one-to-many. Thus, the preimage of $\mathbf{D}_{\mathbf{X}}(\mathbf{P})$ is represented by a set of feasible parameters $\alpha, \beta, \gamma, \delta$ such that all points of $\mathbf{D}_{\mathbf{X}}(\mathbf{P})$ satisfy the inequalities of (3). It means that the preimage is given by a convex polytope in the parameter space [3]. Because all interesting parameters in this paper are translation-invariant, we focus on the three parameters α, β, γ indicating the normal vector of $\mathbf{P}$, distinguished from the intercept δ of $\mathbf{P}$. Further discussion is given in Section 4.

3 Local Geometric Patterns on Digital Planes

3.1 Local Geometric Patterns and Their Linearity

We define a local point set around a point $\boldsymbol{x}$ in $\mathbb{Z}^3$ such that

$$\mathbf{Q}_k(\boldsymbol{x}) = \{\boldsymbol{y} \in \mathbb{Z}^3 : \|\boldsymbol{x} - \boldsymbol{y}\|_\infty \leq k\} \tag{4}$$

where $k = 1, 2, \ldots$. $\mathbf{Q}_k(\boldsymbol{x})$ is a cubical grid-point set whose edge length is $2k+1$. Let us consider that each grid point in $\mathbb{Z}^3$ has a binary value such as either 1 or 0. Such a pattern of binary points in $\mathbf{Q}_k(\boldsymbol{x})$ is called local geometric patterns, abbreviated to LGP hereafter. There are $2^{(2k+1)^3-1}$ different LGP for $\mathbf{Q}_k(\boldsymbol{x})$ providing that the central point $\boldsymbol{x}$ always has a fixed value, such as 1. This indicates that $\boldsymbol{x}$ is considered to be not a background point but an object point.

In this section, we investigate, among these $2^{(2k+1)^3-1}$ LGPs, which LGPs can appear on digital planes. Note that we set binary values of points of $\mathbf{D}_{\mathbf{Q}_k(\boldsymbol{x})}(\mathbf{P})$ to be 1 and those of other points to be 0. This problem is mathematically written as follows. Let $\mathbf{F}$ be a set of points whose values are 1 in $\mathbf{Q}_k(\boldsymbol{x})$. If there is a plane $\mathbf{P}$ such that

$$\mathbf{F} = \{(p, q, r) \in \mathbf{Q}_k(\boldsymbol{x}) \ : \ 0 \leq \alpha p + \beta q + \gamma r + \delta < \omega\}, \tag{5}$$

we say that $\mathbf{F}$ forms a digital plane patch in $\mathbf{Q}_k(\boldsymbol{x})$. Therefore, our problem is solved by looking for all possible $\mathbf{F}$, namely LGP, satisfying (5). Such LGPs are called linear LGPs. Since this problem is considered to be the feasibility of the inequalities of (5) for all $(p, q, r) \in \mathbf{F}$, we check if there are feasible solutions $\alpha, \beta, \gamma, \delta$ for each different $\mathbf{F}$, namely LGP. If they exist, such an LGP appears on digital planes and becomes a linear LGP.

3.2 Linear LGP Generation by Arithmetic Planes

In this paper, however, in order to avoid computing the feasibility test for all $2^{(2k+1)^3-1}$ LGPs of $\mathbf{Q}_k(\boldsymbol{x})$, we take another approach to generate all linear LGPs. Our approach is based on arithmetic planes [7,8], and similar to [4].

From the discussion in Subsection 2.2, we know that there are many possible $\mathbf{P}$ corresponding to a given $\mathbf{D}_{\mathbf{Q}_k(\boldsymbol{x})}(\mathbf{P})$ and that the preimage is represented by a set of feasible parameters $\alpha, \beta, \gamma, \delta$. This implies that, given $\mathbf{D}_{\mathbf{Q}_k(\boldsymbol{x})}(\mathbf{P})$, we can find a corresponding $\mathbf{P}$ with only rational parameters. In addition, the denominators of those rational numbers are bounded by the size of $\mathbf{Q}_k(\boldsymbol{x})$, namely k. It is also known that a digital plane $\mathbf{D}(\mathbf{P})$ with rational slopes is equivalent to an arithmetic plane [7,8], which is defined such that

$$\mathbf{A} = \{(x, y, z) \in \mathbb{Z}^3 \ : \ 0 \leq ax + by + cz + d < w\} \tag{6}$$

where a, b and c are relatively prime integers, and d and w are integers. We call w a thickness of $\mathbf{A}$, and set $w = \max(|a|, |b|, |c|)$, similarly to ω for $\mathbf{D}(\mathbf{P})$. When the thickness is set as mentioned above, $\mathbf{A}$ is called a naive plane [7]. For any $\mathbf{D}_{\mathbf{Q}_k(\boldsymbol{x})}(\mathbf{P})$, it is known that we can find a corresponding $\mathbf{A}$ as its representation.

Based on this fact, we generate all linear LGPs by using naive planes $\mathbf{A}$ instead of digital planes $\mathbf{D}(\mathbf{P})$. Our algorithm mainly consists of the following three steps: **(I)** set parameters $a, b, c, d \in \mathbb{Z}$; **(II)** from those parameters, construct a grid point set $\mathbf{A}$ from (6); **(III)** for each point $\boldsymbol{x} \in \mathbf{A}$, observe the LGP of $\mathbf{Q}_k(\boldsymbol{x})$ for a given k. In the followings, we detail each step of **(I)** and **(II)**.

Parameter Setting for Naive Planes. As we mentioned in Subsection 2.2, we distinguish the three parameters a, b, c indicating a normal vector of $\mathbf{A}$ from the intercept d indicating a translation of $\mathbf{A}$. We are first concerned with the setting of value d. It is known that $\mathbf{A}$ always has grid points $(p, q, r) \in \mathbb{Z}^3$ satisfying the equality of (6) such that

$$ap + bq + cr + d = 0, \tag{7}$$

called leaning points [8]. Chosen a leaning point $\boldsymbol{p} \in \mathbf{A}$, even if we translate $\mathbf{A}$ with a vector $-\boldsymbol{p}$ so that the origin becomes a leaning point, it is certain that such a translation does not influence LGPs on $\mathbf{A}$. We therefore simply set

$$d = 0 \tag{8}$$

so that we consider only naive planes $\mathbf{A}$ one of whose leaning points is the origin.

Concerning the other parameters $a, b, c \in \mathbb{Z}$, we give the following constraints:

$$0 \leq a \leq b \leq c, \ c \neq 0. \tag{9}$$

All $\mathbf{A}$ which do not satisfy (9) can be generated from the $\mathbf{A}$ which satisfy (9) by their rotations around the origin, since we set $d = 0$, and their symmetries with respect to the xy-, yz- and xz-planes. We can also bound c such that

$$c \leq 8k^2 \tag{10}$$

from the size of $\mathbf{Q}_k(\boldsymbol{x})$ [2]. Therefore, once the value of k is given, we can automatically generate a set of all relatively-prime integer triplets (a, b, c), denoted by V_k, by using the Euclidean algorithm, with the constraints (9) and (10).

Constructed Part of a Naive Plane. With (9), we see that the principal projection plane of $\mathbf{A}$ is the xy-plane. Then, it is known that there are maximum $(2k+1)^2$ different LGPs on $\mathbf{A}$ and that all different LGPs can appear in the region which is projected in the principal projection plane, *i.e.*, in the xy-plane, as a $(4k+1) \times (4k+1)$ squared region and whose central point is a leaning point of $\mathbf{A}$, *i.e.*, the origin [11].

Concerning to z-coordinates, thanks to the periodicity of $\mathbf{A}$ [7], we see that the maximum difference of the z-coordinates of any pair of points in the region which is projected in the xy-plane as a $(4k+1) \times (4k+1)$ rectangle does not exceed twice those of the x- and y-coordinates. Therefore, we can set a constructed part of $\mathbf{A}$ in a finite grid space $\mathbf{X}_k$ such that

$$\mathbf{X}_k = [-2k, 2k] \times [-2k, 2k] \times [-4k, 4k]. \tag{11}$$

Algorithm 1. Generation of all linear LGP

```
input  : a size $k$ of $\mathbf{Q}_k(\boldsymbol{x})$
output: a set $T$ of linear LGPs
begin
    initialize a set $T$;
    make a set of integer normal vectors $V_k$ with (9) and (10) ;
    foreach $(a, b, c) \in V_k$ do
        construct a finite grid-point set $\mathbf{A}$ in $\mathbf{X}_k$ from (6) and (11) with (8) ;
        foreach $\boldsymbol{x} \in \mathbf{A} \cap \mathbf{X}_k$ do
            if LGP of $\mathbf{Q}_k(\boldsymbol{x})$ is not included in $T$ then put it in $T$
    return $T$;
end
```

3.3 Algorithm and Results

From the above discussion, we now present Algorithm 1 for generating all linear LGPs of $\mathbf{Q}_k(\boldsymbol{x})$ for a given k. By executing Algorithm 1, we obtain 34 linear LGPs for $k = 1$, similarly 1574 for $k = 2$, 23551 for $k = 3$, and 181735 for $k = 4$, up to translations, rotations and symmetries, thanks to the constraints (9). Figure 1 shows all linear LGPs for $k = 1$.

Remark that our LGP around a point $\boldsymbol{x} = (x, y, z)$ in $\mathbf{A}$ for a given k is slightly different from the $(2k+1, 2k+1)$-cube [11], which is defined as a set of points $\boldsymbol{p} = (p, q, r)$ in $\mathbf{A}$ such that $|x - p| \leq 1$ and $|y - q| \leq 1$ for the case of (9). A linear LGP of $\mathbf{Q}_k(\boldsymbol{x})$ can be smaller than a $(2k+1, 2k+1)$-cube, so that there are less linear LGPs than $(2k+1, 2k+1)$-cubes; for example, there are 40 $(3, 3)$-cubes, called tricubes [4,11], against the 34 linear LGPs for $k = 1$.

3.4 Linear LGPs and Non-linear LGPs

From the above results, we see that there are a few linear LGPs relatively to non-linear ones. However, we have learned from experience that many border points of a digital object have linear LGPs, if its object surface is very smooth. Indeed, this is not difficult to understand, since any local surface patch on a smooth surface can be approximated to a planar surface when the size of the patch becomes small. In other words, even if a point has a linear LGP, we are uncertain whether the point appears on a planar surface or a non-planar surface. Contrarily, if a point has a non-linear LGP, it is certain that the point never appears on a planar surface.

4 Feasible Normal Vectors and Discrete Gaussian Spheres

In the continuous framework, each surface point has a unique normal vector, and a mapping from a surface point to its normal vector on the Gaussian sphere is called a Gaussian image. In [5], an extended Gaussian image is presented as a mapping from a point $\boldsymbol{n}$ on the Gaussian sphere to the area of the surface whose normal vector is $\boldsymbol{n}$. The extended Gaussian images for all n are useful for

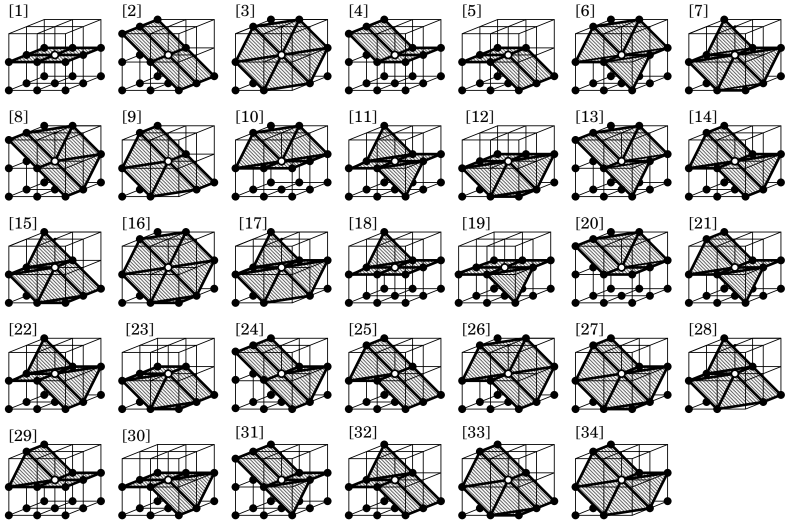

Fig. 1. The 34 linear LGPs for $k = 1$ up to translations, rotations and symmetries, with polyhedral meshes made by a discrete-marching-cube-like method for the 18-neighborhood system [6]

representing surface shapes. In this paper, we need a discrete version of extended Gaussian images, called unified discrete Gaussian images, which we use for planar surface segmentation in the next section. In the discrete framework, each linear LGP has a set of normal vectors, i.e., the preimage, represented by a set of points on the Gaussian sphere. In this section, we first investigate the preimage for every linear LGP, and study how all the preimages divide the Gaussian sphere. Such a divided Gaussian sphere is called a discrete Gaussian sphere. We finally define unified discrete Gaussian images using a discrete Gaussian sphere.

4.1 Feasible Normal Vectors of Each LGP

From the discussion in the previous section, if a surface point $\boldsymbol{x}$ of a digital object has a linear LGP of $\mathbf{Q}_k(\boldsymbol{x})$, we can say that such $\boldsymbol{x}$ is locally linear with the size of $\mathbf{Q}_k(\boldsymbol{x})$. In other words, the value of k indicates the absolute size of a planar surface around $\boldsymbol{x}$. Based on this fact, we therefore calculate normal vectors at $\boldsymbol{x}$ by using (2) from a set of points on a digital plane in $\mathbf{Q}_k(\boldsymbol{x})$.

From each linear LGP, we obtain $\mathbf{D}_{\mathbf{Q}_k(\boldsymbol{x})}(\mathbf{P})$ in $\mathbf{Q}_k(\boldsymbol{x})$, so that for all $(p, q, r) \in \mathbf{D}_{\mathbf{Q}_k(\boldsymbol{x})}(\mathbf{P})$, we obtain a set of linear inequalities (2), namely,

$$0 \leq \alpha p + \beta q + \gamma r + \delta < \omega \tag{12}$$

where $\alpha, \beta, \gamma, \delta \in \mathbb{R}$. Because we assumed (9) for generating the linear LGPs in Section 3, we also consider the similar constraints such that

$$0 \leq \alpha \leq \beta \leq \gamma,\ \gamma \neq 0. \tag{13}$$

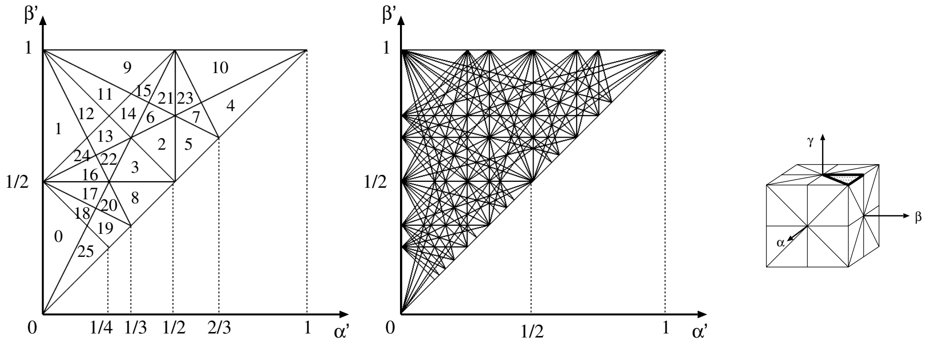

Fig. 2. Normal cells in the parameter space (α', β') for $k = 1$ (left), $k = 2$ (center), and a cubical Gaussian sphere (right)

Note that, thanks to (13), $\omega = \gamma$ in (12) from its definition. Arbitrary choosing pairs of different points $(p_i, q_i, r_i), (p_j, q_j, r_j) \in \mathbf{D}_{\mathbf{Q}_k(\boldsymbol{x})}(\mathbf{P})$, we thus obtain:

$$0 \leq \alpha p_i + \beta q_i + \gamma r_i + \delta < \gamma, \tag{14}$$

$$0 \leq \alpha p_j + \beta q_j + \gamma r_j + \delta < \gamma. \tag{15}$$

Because $\gamma \neq 0$ from (13), we divide both (14) and (15) by γ and substitute α', β' and γ' for α, β and γ such that

$$\alpha' = \frac{\alpha}{\gamma}, \quad \beta' = \frac{\beta}{\gamma}, \quad \gamma' = \frac{\delta}{\gamma}. \tag{16}$$

In order to eliminate γ' by Fourier-Motzkin elimination [14], we then obtain

$$-p_i\alpha' - q_i\beta' - r_i \leq \gamma' < -p_i\alpha' - q_i\beta' - r_i + 1, \tag{17}$$

$$-p_j\alpha' - q_j\beta' - r_j \leq \gamma' < -p_j\alpha' - q_j\beta' - r_j + 1, \tag{18}$$

and finally derive

$$(p_i - p_j)\alpha' + (q_i - q_j)\beta' + r_i - r_j + 1 > 0 \tag{19}$$

for any pair $(p_i, q_i, r_i), (p_j, q_j, r_j) \in \mathbf{D}_{\mathbf{Q}_k(\boldsymbol{x})}(\mathbf{P})$. The solution gives a feasible region which is a convex polygon in the space (α', β'). Remark that all calculations are done by using only integers, *i.e.*, they cause no rounding errors. Figure 2 illustrates that some of those convex polygons share the region in (α', β') and that such common regions make triangular or quadrilateral cells, called normal cells. Note that the constraints (13) make our interesting space (α', β') narrow down to a triangle bounded by linear inequalities $0 \leq \alpha' \leq \beta' \leq 1$. Table 1 shows a set of normal cells which represents a set of feasible normal vectors for each linear LGP of $\mathbf{Q}_1(\boldsymbol{x})$ depicted in Fig. 1.

We can find a figure similar to the left one of Fig. 2 in [11], which is generated by an approach based on 2D Farey series in number theory. It is clear that our

Table 1. Normal cells, illustrated in Fig. 2 (left), corresponding to each linear LGP for $k = 1$, illustrated in Fig. 1, with the constraints (13)

linear LGP for $k = 1$	corresponding normal cells
1	0 25
2	1 9 11 12
3	4 5 7 10 23
4,5	0 1 16 17 18 24
6,17	2 3 4 5 7 8
7	2 3 5 8
8,9	6 9 10 11 14 15 21 23
10,12	8 19 20 25
11	8 17 18 19 20
13,28	2 3 4 5 6 7 9 10 11 12 13 14 15 21 22 23
14	2 3 6 13 14 15 16 21 22 24
15	2 3 6 11 12 13 14 22
16	4 5 7 10 23
18,19	0 18 19 25
20,23	0 1 3 8 12 13 16 17 18 19 20 22 24 25
21,22	3 8 16 17 20 22
24,25	1 9 11 12 13 14 15 24
26,34	2 4 5 6 7 10 21 23
27	2 5 6 7 21 23
29,30	0 17 18 19 20 25
31,32	1 12 13 16 22 24
33	6 9 11 14 15 21

normal cells are related to 2D Farey series since the values of our inputs $p_i - p_j$, $q_i - q_j$ and $r_i - r_j$ of (19) are bounded by the size of $\mathbf{Q}_k(\boldsymbol{x})$. However, our result is slightly different from that in [11]. It is caused by the difference between the definition of linear LGPs and that of $(2k + 1, 2k + 1)$-cubes [11], as we already discussed at the end of Subsection 3.4. The difference can be seen in Fig. 2 (left), such that the normal cells "0" and "10" are not symmetrical with respect to a line of $\alpha' + \beta' = 1$, while the cells in [11] are symmetrical.

4.2 Discrete Gaussian Spheres

The 26 and 910 normal cells in Fig. 2 (left, center) are generated with the constraints (13). We embed these normal cells into the 3D space (α, β, γ) by using (16) with $\gamma = 1$. The triangle surrounded by thick lines in Fig. 2 (right) corresponds to the triangular region which is the union of normal cells in Fig. 2 (left, center). Once the normal cells are embedded into the space (α, β, γ), we make the congruous ones by applying to them 48 transformations of rotations and symmetries of a cube of edge length 2, centered at the origin of the 3D space. We see, in Fig. 2 (right), that there are the 48 triangles on the cube, so that the whole cube contains 1248 and 43680 normal cells for $k = 1, 2$, respectively. Such a cube is called a cubical Gaussian sphere.

We now project normal cells tiled on the cubical Gaussian sphere onto a unit sphere centered at the origin. The unit sphere separated by normal cells is called a discrete Gaussian sphere with respect to k, because the size of normal cells indicates the resolution of digitized normal vectors which are calculated from linear LGPs. Therefore, the resolution of a discrete Gaussian sphere is determined by the size of $\mathbf{Q}_k(\boldsymbol{x})$, *i.e.*, the value of k. In the remainder, we denote

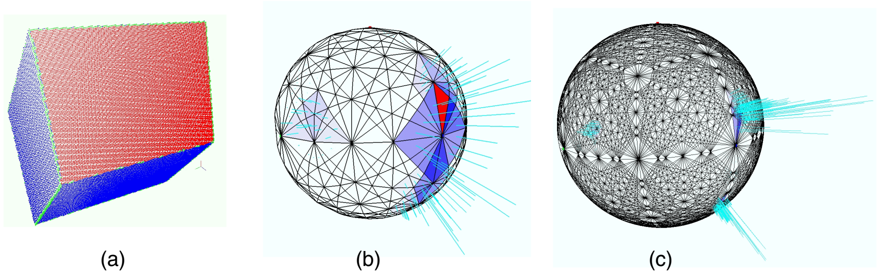

Fig. 3. A synthetic 3D image of a box (a) and its unified discrete Gaussian images for $k = 1$ (b) and $k = 2$ (c). Concerning cell colors on the discrete Gaussian sphere (b,c), the darker the blue cell, the larger the value of $u(c)$, and the red cell has the maximum value. The length of the pale blue needle for each cell c also corresponds to the value of $u(c)$. On a digitized box (a), red and blue points have linear LGPs for $k = 1$, while green points have non-linear LGPs. Note that red points correspond to the red cell.

$\mathbf{G}_k$ the set of all normal cells on the discrete Gaussian sphere with respect to k. Remark that we use only integer or rational numbers to calculate all normal cells, which are related to a cubical Gaussian sphere.

4.3 Unified Discrete Gaussian Images

By using the discrete Gaussian sphere, we give a discrete version of extended Gaussian images [5], called unified discrete Gaussian images. Let us first consider a discrete version of the Gaussian image. Let $\mathbf{V}$ be a set of object surface grid-points. For a point $\boldsymbol{x} \in \mathbf{V}$, we define a discrete Gaussian image $\mathbf{I}_k(\boldsymbol{x})$ as the set of normal cells corresponding to the linear LGP of $\boldsymbol{x}$, if its LGP is linear with respect to k; otherwise, $\mathbf{I}_k(\boldsymbol{x})$ is defined as empty. For each cell $c \in \mathbf{G}_k$, we consider a point subset of $\mathbf{V}$ such that

$$\mathbf{R}(c) = \{\boldsymbol{x} \in \mathbf{V} : c \in \mathbf{I}_k(\boldsymbol{x})\}. \tag{20}$$

We then obtain the number of points in $\mathbf{R}(c)$ for every $c \in \mathbf{G}_k$, called a unified discrete Gaussian image, such that

$$u(c) = |\mathbf{R}(c)|. \tag{21}$$

From the definition, we see that our unified discrete Gaussian image represents a distribution of normal cells of a digital object surface.

Figure 3 shows examples of unified discrete Gaussian images for a digitized box. In the figures, we see that we can extract a set of grid points which belong to a digital plane $\mathbf{D}(\mathbf{P})$ by choosing a “good” cell, for example, a red one. This is based on the following fact; if (α, β, γ) be a normal vector of $\mathbf{D}(\mathbf{P})$, (α, β, γ) is included in the common cell(s) of $\mathbf{I}_k(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbf{D}(\mathbf{P})$.

Algorithm 2. Planar surface segmentation

```
input : a grid-point set $\mathbf{V}$, a size $k$ of LGP, and a minimum point-set size $s$
output: an edge point set $\mathbf{E}$ and planar-surface point sets $\mathbf{S}_i$ for $i = 1, 2, 3, \ldots$
begin
    initialize $u(c) = 0$ and $\mathbf{R}(c) = \emptyset$ for every $c \in \mathbf{G_k}$;
    foreach point $\boldsymbol{x} \in \mathbf{V}$ do
        if $\boldsymbol{x}$ has a linear LGP then
            increment $u(c)$ and add $\boldsymbol{x}$ to $\mathbf{R}(c)$ for all $c \in \mathbf{I}_k(\boldsymbol{x})$;
        else put $\boldsymbol{x}$ in $\mathbf{E}$;
    initialize a label such that $l = 0$;
    repeat
        make a priority queue $D_k$ of cells $c$ with $u(c)$ such that $u(c) \geq s$;
        increment $l$ and initialize $\mathbf{S}_l = \emptyset$;
        set $h$ to be the highest priority cell in $D_k$ and remove it from $D_k$;
        while $u(h) > |\mathbf{S}_l|$ do
            calculate the maximum connected component of $\mathbf{R}(h)$, set to be $\mathbf{C}$;
            if $|\mathbf{C}| > |\mathbf{S}_l|$ then set $\mathbf{S}_l = \mathbf{C}$;
            reset $h$ to be the highest priority cell in $D_k$ and remove it from $D_k$;
        if $|\mathbf{S}_l| \geq s$ then
            forall $c$ such that $u(c) \neq 0$ and $\mathbf{R}(c) \cap \mathbf{S}_l \neq \emptyset$ do
                reset $\mathbf{R}(c) = \mathbf{R}(c) \setminus \mathbf{S}_l$ and $u(c) = |\mathbf{R}(c)|$;
    until $|\mathbf{S}_l| < s$ ;
    return $\mathbf{S}_i$ for $i = 1, 2, \ldots, l - 1$;
end
```

5 Planar Surface Segmentation Using LGPs

In this section, we present an algorithm for planar surface segmentation from a given set of object surface points $\mathbf{V}$. We formulate our problem as follows: we assign each point $\boldsymbol{x} \in \mathbf{V}$ into either an edge set $\mathbf{E}$ or one of surface sets $\mathbf{S}_i$ for $i = 1, 2, \ldots$ such that the points in each $\mathbf{S}_i$ have a common property. In our case, each $\mathbf{S}_i$ corresponds to a connected subset of a digital planar surface, and the points in $\mathbf{E}$ constitute the joints of digital plane patches. We assume that two surface sets have no overlap, namely, $\mathbf{S}_i \cap \mathbf{S}_j = \emptyset$ for any i, j such that $i \neq j$.

5.1 Algorithm

Our method is founded on the following hypothesis; if there is a connected point set $\mathbf{S} \subseteq \mathbf{V}$ such that $\cap_{\boldsymbol{x} \in \mathbf{S}} \mathbf{I}_k(\boldsymbol{x}) \neq \emptyset$, $\mathbf{S}$ may constitute a digital plane patch. Based on this, we propose an algorithm mainly consisting of the following three parts: **(A)** reject all points having non-linear LGPs for a given k and consider them as edge points; **(B)** make a unified discrete Gaussian image $u(c)$, $c \in \mathbf{G}_k$, with the point sets $\mathbf{R}(c)$; **(C)** by using $u(c)$ and $\mathbf{R}(c)$, look for connected point sets, corresponding to planar surfaces, each of whose points having a common normal cell. The parts **(A)** and **(B)** are considered to be pre-processing of the

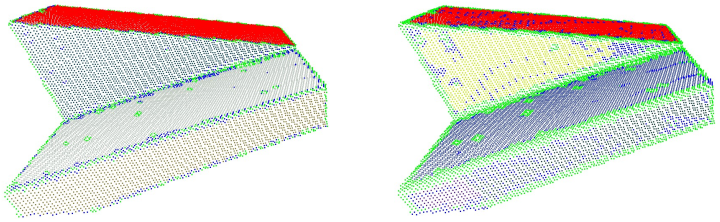

Fig. 4. A result of planar surface detection from a 3D range image of blocks for $k = 1$ (left) and $k = 2$ (right) with $s = 100$. Four (resp. five) planar surfaces are detected for $k = 1$ (resp. $k = 2$) so that each surface is colored with a different color except for light green and blue. Light green points have non-linear LGPs and blue points have linear LGPs but cannot constitute a planar surface whose size is not less than s.

part **(C)** which is the core of the algorithm. We present Algorithm 2, in which **(A)** is achieved in Steps 2 and 2, and **(B)** is achieved from Step 2 to 2.

The core part **(C)**, which starts from Step 2, is a loop of seeking planar surfaces $\mathbf{S}_i$. Each $\mathbf{S}_i$ is a maximally connected point set, whose points have a common normal cell, as shown in Steps from 2 to 2. Once we find $\mathbf{S}_i$, we check the size of $\mathbf{S}_i$ in Step 2, and if $|\mathbf{S}_i| \geq s$, we remove all points of $\mathbf{S}_i$ from every point set $\mathbf{R}(c)$ and sometimes modify $u(c)$ in Step 2. After such modification and incrementing i, we seek a new $\mathbf{S}_i$. For finding each $\mathbf{S}_i$, we look for the maximum connected component $\mathbf{C}$ of each $\mathbf{R}(c)$ and then set $\mathbf{S}_i$ to be the maximum among all $\mathbf{C}$. In order to reduce the frequency of calculation of connected components which is a global operation, we make a priority queue D_k of cells with $u(c)$ in Step 2. We then repeat dequeue of a cell h from D_k to obtain the maximum connected component $\mathbf{C}$ of $\mathbf{R}(h)$ in Step 2. Comparing the size of $\mathbf{C}$ with the maximum among those of other cells which are already dequeued from D_k, we finally obtain the currently maximum point set $\mathbf{S}_l$ in Step 2. Note that this loop is repeated until $u(h)$ is not more than the size of $\mathbf{S}_l$ as described in Step 2. For calculating the maximum connected component of $\mathbf{R}(h)$, we apply a simple method based on a depth-first strategy by using a queue [7]. The time complexity is linear with respect to $u(h)$, namely, $O(n)$, where n is the number of grid points in $\mathbf{V}$, in the worst case. Thus, the time complexity of the whole algorithm is $O(ln)$, where l is the number of segmented regions.

5.2 Experimental Results

Figure 4 shows the results of planar surface segmentation from a 3D range image of blocks. We see that four (resp. five) planar surfaces are detected for $k = 1$ (resp. $k = 2$), even if the range image contains noise. We see that one of planes for $k = 1$ is split into two for $k = 2$ because of the different normal-cell resolutions between $k = 1$ and $k = 2$. Remark that non-linear LGPs appear around edges and ridge lines of blocks, and also appear in faces of blocks because of noise in the range image. We also see that there are more non-linear LGPs for $k = 2$ than $k = 1$. In other words, the higher k, the more sensitive the result. Linear LGPs whose corresponding normal cells are not shared by those of the neighboring LGPs mostly appear beside non-linear LGPs.

6 Conclusion

We present a discrete version of the hybrid method for planar surface segmentation from a 3D grid-point set. We first propose an algorithm to generate all linear LGPs for $k = 1, 2, \ldots$ by using the arithmetic plane properties. Based on those linear LGPs, we also introduce new notions of discrete Gaussian spheres and unified discrete Gaussian images, in order to process normal vectors in the discrete-geometry framework. By using those discrete notions, we succeed in detecting an edge-point set and planar-surface point sets. With our experiments, we show that our method is robust against not only quantization errors but also noise. As our method gives a rough segmentation result, it may be useful to obtain an initial segmentation before applying plane recognition.

For future work, the comparison with conventional methods in both computer vision and discrete geometry in terms of segmentation results and computational costs are left. In addition, we need further work on surface segmentation for non-polyhedral objects.

References

1. Besl, P.J., Jain, R.C.: Segmentation through variable-order surface fitting. IEEE Transactions on PAMI 10(2), 167–192 (1988)
2. Buzer, L.: A composite and quasi linear time method for digital plane recognition. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 331–342. Springer, Heidelberg (2006)
3. Coeurjolly, D., Sivignon, I., Dupont, F., Feschet, F., Chassery, J.-M.: On digital plane preimage structure. Discrete Applied Mathematics 151(1–3), 78–92 (2005)
4. Debled-Rennesson, I.: Etude et reconnaissance des droites et plans discrets. Ph.D. thesis, Université Louis Pasteur, Strasbourg (1995)
5. Horn, B.K.P.: Extended Gaussian images. Proceedings of the IEEE 72(12), 1671–1686 (1984)
6. Kenmochi, Y., Imiya, A.: Combinatorial boundary of a 3D lattice point set. Journal of Visual Communication and Image Representation 17(4), 738–766 (2006)
7. Klette, R., Rosenfeld, A.: Digital Geometry: Geometric Methods for Digital Picture Analysis. Morgan Kauffmann, San Francisco (2004)
8. Reveillès, J.-P.: Combinatorial pieces in digital lines and planes. In: Vision Geometry IV. SPIE, vol. 2573, pp. 23–34 (1995)
9. Sivignon, I., Dupont, F., Chassery, J.-M.: Discrete surfaces segmentation into discrete planes. In: Klette, R., Žunić, J. (eds.) IWCIA 2004. LNCS, vol. 3322, pp. 458–473. Springer, Heidelberg (2004)
10. Stamos, I., Allen, P.K.: 3D model construction using range and image data. In: Proceedings of IEEE Conference on CVPR, vol. 1, pp. 531–536 (2000)
11. Vittone, J., Chassery, J.-M.: (n, m)-cubes and Farey nets for naive planes understanding. In: Bertrand, G., Couprie, M., Perroton, L. (eds.) DGCI 1999. LNCS, vol. 1568, pp. 76–87. Springer, Heidelberg (1999)
12. Yokoya, N., Levine, M.D.: Range image segmentation based on differential geometry: a hybrid approach. IEEE PAMI 11(6), 643–649 (1989)
13. Zhao, D., Zhang, X.: Range-data-based object surface segmentation via edges and critical points. IEEE Transactions on IP 6(6), 826–830 (1997)
14. Ziegler, G.M.: Lectures on Polytopes. Springer, New York (1998)