JAIST Repository
https://dspace.jaist.ac.jp/

Title	Combinatorial and Geometric Problems Related to Digital Halftoning
Author(s)	Asano, Tetsuo; Katoh, Naoki; Obokata, Koji; Tokuyama, Takeshi
Citation	Lecture Notes in Computer Science, 2616: 101-106
Issue Date	2003
Type	Journal Article
Text version	author
URL	http://hdl.handle.net/10119/4914
Rights	This is the author-created version of Springer, Tetsuo Asano, Naoki Katoh, Koji Obokata and Takeshi Tokuyama, Lecture Notes in Computer Science, 2616, 2003, 101-106. The original publication is available at www.springerlink.com, http://dx.doi.org/10.1007/3-540-36586-9_4
Description	

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

Combinatorial and Geometric Problems Related to Digital Halftoning

Tetsuo Asano[1], Naoki Katoh[2], Koji Obokata[1], and Takeshi Tokuyama[3]

[1] JAIST, Tatsunokuchi, Japan,
[2] Kyoto University, Kyoto, Japan,
[3] Tohoku Iniversity, Sendai, Japan.

Abstract. Digital halftoning is a technique to convert a continuous-tone image into a binary image consisting of black and white dots. It is an important technique for printing machines and printers to output an image with few intensity levels or colors which looks similar to an input image. The purposes of this paper are to reveal that there are a number of problems related to combinatorial and computational geometry and to present some solutions or clues to those problems.

1 Introduction

The quality of color printers has been drastically improved in recent years, mainly based on the development of fine control mechanism. On the other hand, there seems to be no great invention on the software side of the printing technology. What is required is a technique to convert a continuous-tone image into a binary image consisting of black and white dots that looks similar to the input image. Theoretically speaking, the problem is how to approximate an input continuous-tone image by a binary-tone image. Since this is one of the central techniques in computer vision and computer graphics, a great number of algorithms have been proposed (see, e.g., [6, 8, 9, 17]) with several theoretical results by the authors [1–5]. The purpose of this paper is to reveal that some notions on combinatorial and computational geometry such as Voronoi diagram, discrepancy, and dispersion are related to digital halftoning. Based on those notions we shed light on digital halftoning from different directions.

2 Known Basic Algorithms

Throughout the paper we put the following assumptions to simplify the discussion. We take as an input image an $N \times N$ real-valued matrix $A = (a_{ij})$, $0 \leq a_{ij} \leq 1$ for each (i, j) and output a binary matrix $B = (b_{ij})$ of the same size. Usually, black has the intensity level 0 while white has 1.

1	33	9	41	3	35	11	43
49	17	57	25	51	19	59	27
13	45	5	37	15	47	7	39
61	29	53	21	63	31	55	23
4	36	12	44	2	34	10	42
52	20	60	28	50	18	58	26
16	48	8	40	14	46	6	38
64	32	56	24	62	30	54	22

Fig. 1. 8×8 dither matrix by Bayer [6].

For color images, we iterate the same halftoning process three times for each of R (Red), G (Green), and B (Blue) components.

Since digital halftoning is a widely used technique, a great number of algorithms have been proposed so far (refer to the books [10, 17]). First we briefly describe several representative halftoning algorithms with their variations of algorithmic flavor.

2.1 Simple Thresholding

Given an $N \times N$ array A of real numbers between 0 and 1, we wish to construct a binary array B of the same size which looks similar to A, where entry values represent light intensity levels at corresponding locations. The most naive method for obtaining B is simply to binarize each input value by a fixed threshold, say 0.5. It is simplest, but the quality of the output image is worst since any uniform gray region becomes totally white or totally black. The most important is how to represent intermediate intensities.

2.2 Ordered Dither

Instead of using a fixed threshold over an entire image, this method uses different thresholds. A simple way of implementing this idea is as follows: We prepare an $M \times M$ matrix of integers from 1 to M^2. This matrix (dither array) is tiled periodically to cover the image. Each pixel in the image is compared with the corresponding threshold from the dither array to decide whether a dot will be placed at that location. Fig. 1 shows the dither matrix given by Bayer [6].

2.3 Error Diffusion

The dither algorithm is designed to preserve the average intensity level between input and output images. There is another standard algorithm called "error diffusion" that also possesses the same property by propagating the quantization errors to unprocessed neighboring pixels according to some fixed ratios. More precisely, pixels are processed in a raster order, from left to right and top to bottom. Each pixel level is compared with a fixed threshold, 0.5 and round it up if it is greater than or equal to the threshold and round it down otherwise. The quantization error caused by the rounding is diffused over the unprocessed pixels around it with fixed ratios. For example, if a pixel level is 0.7, it is rounded up to 1 and the error -0.3 is diffused to the unprocessed pixels nearby. The ratios suggested by Floyd and Steinberg in their paper [8] are shown in Fig. 2:

$$\begin{array}{ccc} & \bullet & 7/16 \\ 3/16 & 5/16 & 1/16 \end{array}$$

Fig. 2. Diffusion ratios in Error Diffusion by Floyd and Steinberg.

This method certainly preserves the average intensity level because the rounding error is distributed to neighboring pixels. When the process terminates, the difference between the sums of intensity levels in the input and output images is at most 0.5.

This method not only preserves the average intensity level but also gives excellent image quality in many cases, but it tends to produce visible artifacts in an area of uniform intensity, which are caused by the fixed error-diffusing coefficients.

Fig. 3. Output images: Simple thresholding, ordered dither, and error diffusion (color images).

Fig. 3 compares the three algorithms, simple thresholding, ordered dither and error diffusion, by their output images.

3 Variation of Known Algorithms with Related Problems

3.1 Variation of Simple Thresholding

The serious drawback of the simple thresholding is poor expression of intermediate intensity due to its independent process at each pixel and its use of a fixed threshold. One of the method to improve the expression is to use random thresholds. Precisely, we generate white gaussian noise over an input image and use the noise as threshold. This method is considered as variation of ordered dither with a dither matrix defined by random numbers. Thus, theoretically speaking, the expected average intensity level of the output image is expected to be equal to that of input image.

The same idea is popular in randomized algorithms under a different name, i.e., randomized rounding [14, 16], in which a real number x, $0 \leq x \leq 1$, is rounded up with probability x. It is one of the standard techniques in randomized algorithms.

3.2 Variation of Ordered Dither Algorithm

The previous subsection described a rounding algorithm using variable random numbers as thresholds to generalize the simple thresholding that uses one fixed threshold. We can use a carefully designed table of random numbers instead of generating a random number for each pixel. Dither matrix corresponds to this table of random numbers. Small table size tends to lead to disadvantage of visible artifacts. So, the largest possible table size would be better. In fact, there is an algorithm along this idea, which is known as a blue-noise mask algorithm [12, 18, 19] in general. This algorithm uses a large dither matrix (blue-noise mask) of size, say 256×256.

Properties of Dither Matrix

The performance of the ordered dither algorithm heavily depends on a dither matrix used. We have known how to construct the dither matrix. Then what is a merit to use this dither matrix? In other words, does it optimize anything? If the purpose is only to distribute numbers 1 through 2^{2k} over the $2^k \times 2^k$ matrix, there are a number of different ways. Imagine an artificial image of gradually increasing intensity from left to right.

During the transition from dark to bright, the number of white dots should gradually increase. This means that for any number i between 1 and 2^{2k} those entries having numbers greater than i must be as uniformly distributed as possible over the dither matrix. The uniformity can be measured in several different ways. One measure is based on the ratio between the smallest and largest diameters of empty circles containing no point in their interior but at least two points on the circles. The smallest empty circle is attained by the minimum pairwise distance. The largest one is defined by the largest circle passing through three points while containing no point in its interior with its center lying in the convex hull of the point set. Another possible measure is based on the notion of "discrepancy" which is related to the difference between the area and the relative number of white dots.

The above regular grid-like construction of the dither matrix is optimal in the former measure since it is constructed under the notion of incremental Voronoi insertion. An optimal dither matrix under the former measure is designed as follows. Before construction we have to note that dither matrix is used to cover an entire image by repeatedly arranging the matrix. First we choose an arbitrary entry, say, the upper left corner of the matrix, to assign number 1. Because of the periodicity, it means that we have placed points numbered 1 on regular grids $(8i, 8j), i, j = 0, 1, \ldots$. The entry 2 must be placed at a grid point farthest from the points numbered 1. Such a place coincides with a Voronoi vertex of the Voronoi diagram for the set of points numbered 1. Similarly, the location of the entry 3 should be chosen among the Voronoi vertices for the Voronoi diagram of the set of points numbered 1 or 2. This strategy is called "incremental Voronoi insertion" which is rather easy to be implemented. If we resolve ties appropriately we obtain the dither matrix.

Unfortunately, this dither matrix is not good enough in practice. What is wrong? The measure may be wrong. That is, the measure based on the ratio between the minimum pairwise distance and the diameter of the maximum empty circle may not be good enough to reflect the uniformity of point distribution. The latter measure based on the discrepancy suggested above seems to be more promising. In the measure we take a number of regions. If points are uniformly distributed, the point density is roughly the same in each such region. In the discrepancy measure we can take regions of arbitrary shapes. The former measure based on the minimum pairwise distance is roughly equal to the discrepancy measure for a family of circular regions. In this sense the discrepancy measure is a generalization of the former measure.

To define the discrepancy measure, we introduce a family $\mathcal{F}$ of regions over an image. For each region R in $\mathcal{F}$, let $A(R)$ denote the area of R and $card(R)$ denote the number of points in R. Then, we take the difference

$$D(R) = |n \cdot A(R) - card(R)|,$$

as the discrepancy for the region R, assuming that the area of the whole image is 1.

Consider a regular pattern in which n points are placed in a $\sqrt{n} \times \sqrt{n}$ grid. Take a rectangular region R defined by two rows of points. Then, the area of the rectangle is $(1/\sqrt{n}) \times 1 = 1/\sqrt{n}$. If we locate the rectangle so that the two sides exactly coincide with two rows of points, we have $card(R) = 2\sqrt{n}$. Otherwise, it contains only one of rows of points, and so $card(R) = \sqrt{n}$. Thus, we have $D(R) = |n/\sqrt{n} - 2\sqrt{n}| = \sqrt{n}$ in the former case and $D(R) = |n/\sqrt{n} - \sqrt{n}| = 0$ in the latter case. In fact, we can prove that the maximum value of $D(R)$ is $O(\sqrt{n})$. Furthermore, it is known that it remains $O(\sqrt{n})$ when n points are randomly distributed. However, there are deterministic algorithms which achieves the discrepancy $O(\log n)$. Refer to the textbooks on discrepancy by Chazelle[7] and Matoušek[11].

Rotation of Dither Matrix

Another related problem comes from the human perception. An interesting feature of human perception is that horizontal and vertical patterns are more sensitive to human eyes than skewed patterns [15]. This fact suggests us of rotating a dither matrix. Then, the problem is how to design such a rotated pattern consisting of M^2 elements. This is not so easy since a rotated pattern must be tiled to cover the entire image and the area (number of entries) is fixed. Fig. 4 illustrates how a rotated dither matrix covers the entire plane.

We shall explain how to design a pattern which satisfies the following conditions:

(1) area condition: The rotated matrix must have the same number of entries (or grid points) as that of the original matrix, and those grid points form a connected cluster without any hole.

(2) tiling condition: The rotated matrix must be tiled to cover the entire grid plane, that is, the entire plane must be tiled without any gap by repeated placements of the same pattern.

(3) angle condition: The rotated matrix must be bounded by four digital lines segments. The angle of those segments from the axes should

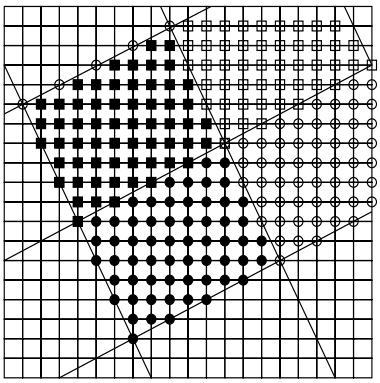

Fig. 4. Tiling the entire grid by a pattern.

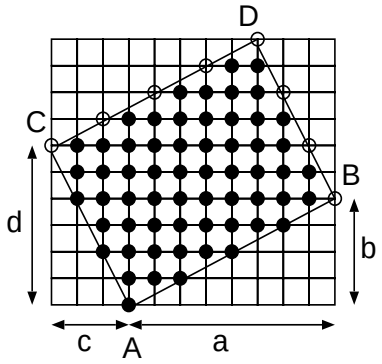

Fig. 5. A tiling pattern $R = ABCD$ and four parameters a, b, c, d defining it.

be close enough to a given angle. Furthermore, the angle between two such segments should be almost perpendicular.

The most important observation behind the scheme for achieving a rotation is the following Pick's theorem [13].

[Pick's Theorem] The area of any simple polygon P in a grid (not necessarily convex) whose vertices are lattice points is given by

$$\text{area}(P) = L_{in}(P) + L_{bd}(P)/2 - 1,$$

where $L_{in}(P)$ denotes the number of grid points in the interior of the polygon P and $L_{bd}(R)$ that of grid points on the boundary.

Our objective is to design a rotated square region R consisting of M^2 grid points rotated approximately by an angle θ. We have four vertices

denoted by A, B, C and D, as shown in Fig. 5. Among the four vertices only the vertex A is included in the rotated pattern. Since this is a tiling pattern, the other three vertices become the positions at which the A corner of the pattern R is located. The rotated square R has four sides, AB, AC, BD and CD. The grid points on the lower sides AB and AC are included into the rotated pattern R while those grid points on the upper sides BC and CD are not. Here note that by symmetry the number of grid points on the lower sides is equal to that of grid points on the upper sides. See Fig. 5 for illustration.

Then, the number of grid points included in the rotated pattern R is given by the sum of the number of grid points lying in the interior of the rotated square $R = ABCD$, half the number of grid points on the four sides excluding the vertices, and 1 for the vertex A. Thus, by the Pick's theorem, the number of grid points in the rotated square pattern R is

$$\begin{aligned} & L_{in}(R) + \frac{1}{2}(L_{bd}(R) - 4) + 1 \\ = & L_{in}(R) + \frac{1}{2}L_{bd}(R) - 1 = area(R). \end{aligned}$$

Now, given a size of a rotated pattern (the number of grid points) and an angle θ, we can construct such a rotated pattern as follows:

Designing a Rotated Pattern

(1) Find four integers a, b, c, and d such that

$$\begin{aligned} ad + bc &= M^2, \\ \frac{b}{a} \simeq \frac{c}{d} &\simeq \tan\theta. \end{aligned}$$

(2) Determine a quadrangle $R = ABCD$ such that

1. The bottom, right, left, and top vertices of R are A, B, C, and D.
2. The coordinates of the vertices B, C, D are determined by $(x_A + a, y_A + b)$, $(x_A - c, y_A + d)$, $(x_A - c + a, y_A + d + b)$, respectively.
3. The grid points on the lower side are included in R.
4. The grid points on the upper side are not included in R.
5. Among the four vertices, only the bottom vertex A is included in R.

Lemma 1. *The rotated pattern R designed above satisfies the three conditions listed above.*

Proof. **Area Condition** is satisfied since the area of the rotated pattern R is

$(a+c)(b+d) - ab - cd = ad + bc.$

Tiling Condition: The pattern can tile the entire grid. When we translate R so that the A corner coincides with the location of the vertex B, the side AC of the translated pattern coincides with the side BD of the pattern in the original location. By the definition of the rotated pattern, they coincide with each other and those grid points on the side are included only in the translated pattern. It is just the same for the other sides. Finally, there is no collision of vertices since we have chosen only one vertex among the four vertices.

Angle condition: We can choose the four parameters a, b, c, d so that the slopes of the sides AB and CD are roughly equal to $\tan\theta$ and those of the sides AC and BD are roughly equal to $\tan(\theta - \pi/2)$. Thus, we can choose the best possible parameter values among those values satisfying the area condition.

Blue-noise Mask: a Huge Dither Matrix

One way to remove the artifact texture pattern of the Ordered dither algorithm is to rotate the dither matrix. There is another way. Just use a huge dither matrix of size, say, 256×256. If we carefully design such a huge dither matrix, artifact textures are not visible anymore. The problems are how to design such a huge dither matrix and the large storage requirement.

Such a huge dither matrix is generally referred to as a blue-noise mask. Important is to remove periodicity. Consider a dither matrix of size 256×256. When we have 256 intensity levels, each number between 1 and 256 appears 256 times in the matrix. For each number p between 1 and 256, those entries numbered 1 through p should be distributed as uniformly as possible. A desired pattern is not a regular one but somewhat random-looking pattern as is explained concerning discrepancy. There are several ways to incorporate randomness. One such method is the one called "void-and-cluster" algorithm [19].

The algorithm starts with a random distribution of points and gradually tries to reform the pattern for uniform distribution. There are two factors to break uniformity: cluster parts in which many points are located closely to each other and void parts in which points are sparsely distributed. An idea to achieve uniform distribution is to move a point in a cluster to the center of a void.

Such an operation is well supported in computational geometry. Given n points, the Voronoi diagram is constructed in $O(n \log n)$ time. When a point is surrounded by many points, its associated Voronoi region tends to be small. Thus, cluster parts are found by checking areas of Voronoi regions. On the other hand, void parts correspond to sparse parts. Such locations are found as centers of large empty circles in which no point is contained. A largest empty circle can be found in linear time in two ways, one based on linear programming and the other on randomization.

3.3 Dispersion Problem

Now it finds that the problem of designing a good blue-noise mask is closely related to the following combinatorial problem.

Dispersion Problem We want to insert a predermined number of points one by one as much uniformly as possible in some given domain at any instance. The uniformity is measured by the maximum ratio of the maximum gap over the minimum one. When the maximum and minimum gaps after inserting k points are denoted by G_k and g_k, respectively, the ratio r_k is defined by $r_k = G_k/g_k$. The objective here is to minimize the maximum ratio $R_n = \max(r_1, r_2, \ldots, r_n)$, which is referred to as the dispersion of the point sequence.

There may be several different manners to define a gap. In the d-dimensional space we define it by the radius of a ball with its center being in the domain which contains $d+1$ points on the surface but no point in its interior. The problem here is to find upper and lower bounds on the maximum ratio R_n in each dimension.

General Approximation Algorithm We assume that a domain is given by a convex polyhedron and its vertices are included as an initial set of points. Then, we add points one by one. A simple way to insert points uniformly is a so-called incremental Voronoi insertion, which inserts a point at an interior vertex of a current Voronoi diagram that has the largest clearance around it.

It is not so hard to check the performance of the algorithm. In 1-D, starting with a unit interval $[0,1]$ with an initial set $\{0,1\}$, we put a new point at the center point of the the current largest gap $[0,1]$. Thus, a sequence of points generated is $\{p_1 = 1/2, p_2 = 1/4, p_3 = 3/4, p_4 = 1/8, \ldots\}$. Then, starting with a gap $G_0 = g_0 = 1$, the largest and smallest gaps when we inserted the k-th point are $G_k = 1/2^{\lfloor \lg(k+1) \rfloor}$ and $g_k =$

$1/2^{\lfloor \lg k \rfloor + 1}$, respectively. Thus, the ratio r_k is 1 if $k = 2^i - 1$ for some integer i and 2 otherwise. Thus, the maximum ratio, i.e., the dispersion of the 1-D Voronoi insertion, is 2.

What about the 2-d case? We start with a unit square with an initial point set $\{(0,0),(0,1),(1,0),(1,1)\}$. Then, we do the incremental Voronoi insertion. It proceeds similarly as the one-dimensional case. That is, the ratio is either 1 or $\sqrt{2}$. Thus, the maximum ratio is $\sqrt{2}$.

One-Dimensional Case Our domain here is a unit interval $[0,1]$. The two extremal points 0 and 1 are assumed to be included in the set. We can show that there is a strategy better than the incremental Voronoi insertion.

As an exercise, let us consider the case when we insert 3 points. Unlike the incremental Voronoi insertion, we put the first point p_1 so that the unit interval is split unevenly. Then, we put the second point p_2 to split the longer interval. Now we split the current longest interval into two by putting the third point p_3. This process is represented by a binary tree rooted at the unit interval. It is followed by two intervals x_1 and x_2, where $x_1 + x_2 = 1$ with $x_1 > x_2$. Then, x_1 has branches to x_3 and x_4 with $x_3 + x_4 = x_1$ and $x_3 \geq x_4$. The node x_2 is also followed by two nodes x_5 and x_6 such that $x_5 + x_6 = x_2$ and $x_5 \geq x_6$.

Then, the ratios are $r_1 = x_1/x_2$, $r_2 = x_2/x_4$, and $r_3 = x_3/x_6$. Since the intervals $x_3, \ldots, x_6$ are not split anymore, the partition of x_1 into x_3 and x_4 and that of x_2 into x_5 and x_6 should be bisections at their center points to minimize the ratios, that is, $x_3 = x_4$ and $x_5 = x_6$. Now, let us denote x_5 and x_3 by y_1 and y_2, respectively. Then, $x_2 = 2y_1$ and $x_1 = 2y_2$. Therefore, $r_2 = x_2/x_4 = 2y_1/y_2$ and $r_3 = x_3/x_6 = y_2/y_1$. Thus, the maximum ratio R_3 is minimized when r_2 and r_3 are equal, and it is given by

$$R_3 = \sqrt{r_2 \cdot r_3} = \sqrt{\frac{2y_1}{y_2}\frac{y_2}{y_1}} = \sqrt{2}. \tag{1}$$

We can show that this bound is optimal, that is, there is no sequence of three points to achieve a better ratio. Assume w.l.o.g. that $0 < p_1 \leq 1/2$. Then, the ratio r_1 is given by $(1 - p_1)/p_1$, which is at least $\sqrt{2}$. Thus, $1/2 \geq p_1 \geq \sqrt{2} - 1$. We have to choose p_2 in the interval $[p_1, 1]$ to have a ratio at most r_1. So, again w.o.l.g. we assume that $p_2 - p_1 \geq (1-p_1)/2$, and thus $p_2 \geq (1+p_1)/2$. Now, the minimum gap is between p_2 and 1, and thus the ratio r_2 is given by $p_1/(1-p_2)$, which is bounded by $\sqrt{2}$. This leads to $p_2 \leq 1 - p_1/\sqrt{2}$. Combining the results, we have $p_2 \geq (1+p_1)/2 \geq \sqrt{2}/2$,

and $p_2 \leq 1 - p_1/\sqrt{2} \leq 1 - (\sqrt{2}-1)/\sqrt{2} = \sqrt{2}/2$. Thus, p_2 must be $\sqrt{2}/2$. It is also seen that p_1 must be $\sqrt{2}-1$. So, whatever p_3 is, the ratio R_3 cannot be better than $\sqrt{2}$.

We can generalize the result above in the following forms.

Lemma 2. *There is a sequence of real numbers* $(p_1, p_2, \ldots, p_n)$ *in the unit interval* $[0,1]$ *with the dispersion*

$$R_n = 2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}. \tag{2}$$

Proof. First we consider the case when n is an even number, that is, $n = 2k$ for some integer k. The strategy to insert points is to bisect a current longest interval each time unevenly if the resulting intervals will be further partitioned and evenly otherwise. We rename the last $2k$ intervals as follows:

$$\begin{aligned} x_{4k-1} = x_{4k} &\longrightarrow y_1, \\ x_{4k-3} = x_{4k-2} &\longrightarrow y_2, \\ \vdots \quad \vdots & \\ x_{2k+1} = x_{2k+2} &\longrightarrow y_k. \end{aligned}$$

Then, the ratios are given by

$$r_{2k} = \frac{x_{2k}}{y_1},\ r_{2k-1} = \frac{2y_1}{y_2},\ r_{2k-2} = \frac{2y_2}{y_3},\ \cdots, r_k = \frac{2y_k}{x_{2k}}.$$

The maximum ratio $R_n = \max\{r_{2k}, r_{2k-1}, \ldots, r_k\}$ is minimized when $r_{2k} = r_{2k-1} = \cdots = r_k$. Hence, we have

$$R_n = \left(\frac{x_{2k}}{y_1} \cdot \frac{2y_1}{y_2} \cdot \frac{2y_2}{y_3} \cdots \frac{2y_k}{x_{2k}}\right)^{1/(k+1)} = 2^{k/(k+1)},$$

which is equal to $R_n = 2^{\lfloor n/2 \rfloor / (\lfloor n/2 \rfloor + 1)}$.

We have a similar proof when n is an odd number, that is, $n = 2k+1$ for some k. In this case we have intervals x_1 through x_{4k+2}. We set

$$\begin{aligned} x_{4k+1} = x_{4k+2} &\longrightarrow y_1, \\ x_{4k-1} = x_{4k} &\longrightarrow y_2, \\ \vdots \quad \vdots & \\ x_{2k+1} = x_{2k+2} &\longrightarrow y_{k+1}. \end{aligned}$$

Then, the ratios are as follows:

$$r_{2k+1} = \frac{y_{k+1}}{y_1},\ r_{2k} = \frac{2y_1}{y_2},\ r_{2k-1} = \frac{2y_2}{y_3},\ \cdots, r_{k+1} = \frac{2y_k}{y_{k+1}}.$$

The maximum ratio $R_n = \max\{r_{2k+1}, r_{2k}, \ldots, r_{k+1}\}$ is minimized when they are all equal. Hence, we have

$$R_n = (\frac{y_{k+1}}{y_1} \cdot \frac{2y_1}{y_2} \cdot \frac{2y_2}{y_3} \cdots \frac{2y_k}{y_{k+1}})^{1/(k+1)} = 2^{k/(k+1)},$$

which is again equal to $R_n = 2^{\lfloor n/2 \rfloor/(\lfloor n/2 \rfloor+1)}$.

3.4 Formulation by Integer Linear Programming

The problem of optimally distributing k points over a square grid of area n can be formulated as an Integer Linear Program. The problem $P(n,k)$ is, given a grid of area n, to choose $k(\leq n)$ lattice points on the grid so that the minimum pairwise distance is maximized. To solve this problem we consider a slightly different problem $P'(n,d)$; Given a grid of area n and a real number d, choose as many lattice points in the grid as possible so that their minimum pairwise distance is greater than d. Since there are only $O(n^2)$ possible values for the pairwise distances, if we solve the problem $P'(n,d)$ for $O(\log n)$ different discrete values of d we can obtain a solution to the original problem $P(n,k)$.

To solve a problem $P'(n,d)$ using an integer linear program, we define a binary variable $x_{i,j}$ for each lattice point (i,j) in the grid of area n, which is 1 if we choose the corresponding lattice point and 0 otherwise. Then, the problem is to maximize the sum $\sum x_{i,j}$, the number of grid points chosen, under the constraint that there is no pair of points with distance $\leq d$. The corresponding set of linear inequalities are obtained if we enumerate all possible pairs of lattice points with distance $\leq d$.

In the continuous plane it is rather easy to define a region that contains every possible vector of length at most d. Take two points a and b of distance d and draw two circles of radius d centered at a and b. Then, let c be one of the intersections of the two circles and draw a circle of radius d centered at c. Now, the intersection of these three disks is the region required that contains every possible vector of length at most d in it. Note that the three points form a regular triangle. See Fig.6 for illustration.

In our case we want to have a region that contains all possible integer vectors of length at most d, where an integer vector is defined by a pair

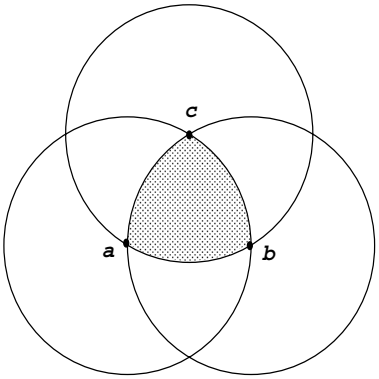

Fig. 6. A region containing every possible vector of length at most d.

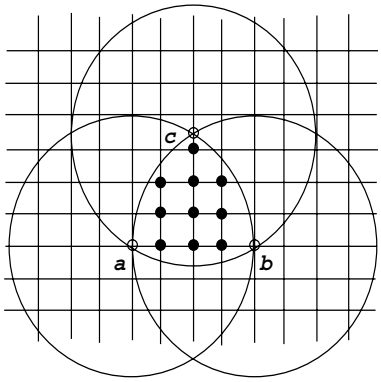

Fig. 7. A discrete region defined by three circles.

of lattice points. Unfortunately, the above method by three circles does not apply to the discrete case. Consider the case of $d = 4$. We take two points a and b of distance 4. Then, the corresponding circles intersect at a point c that is just in the middle of the two lattice. If we draw a circle of radius 4 centered at c, the resulting region does not contain a vertical integer vector of length 4 in it. See Fig.7.

To resolve the difficulty we use at most two regions instead of one. Given a distance d, we take its integer part $d^* = \lfloor d \rfloor$. Then, we take a horizontal integer vector (a, b) of length d^*. Let R_h be a set of all lattice points in the intersection of the half plane above the line through a and b and the two disks of radius d centered at a and b. Similarly we define a

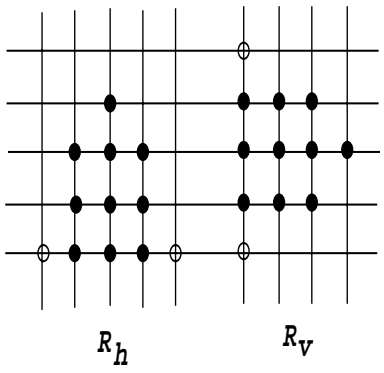

Fig. 8. Two discrete regions containing every possible vector of length at most 4.

set R_v by rotating R_h by 90 degrees around the point a. Fig.8 shows two such regions for $d = 4$.

It is easy to prove that every integer vector of length at most d is contained in one of the regions. By the construction, if we take the lattice point a as an initial point, R_h contains every integer vector of angle in $[0, \pi/3]$. By symmetry, it also contains one of angle in $[\pi, 4\pi/3]$. If we fix the lattice point b as one of endpoints of an integer vector, R_h contains every integer vector of angle in $[2\pi/3, \pi]$ or $[5\pi/3, 2\pi]$. We have the same observation for the region R_v, which covers the angular intervals $[\pi/6, \pi/2], [7\pi/6, 3\pi/2], [\pi/2, 5\pi/6]$, and $[3\pi/2, 11\pi/6]$.

4 Conclusion

In this paper we have shown that several problems on discrete and computational geometry are related to digital halftoning. There are a number of open problems. Many of them are related to conversion from the continuous plane to discrete plane. One such example is a discrete dispersion problem in which points must be laid at some lattice points. Many things are left unknown for such discrete versions of the problems.

Acknowledgment

The first author expresses his sincere thanks to Naoki Fujikawa (former JAIST student), Tomomi Matsui (Tokyo Univ.), Hisao Tamaki (Meiji Univ.), Hiroshi Nagamochi (Toyohashi Univ. Tech.), and Nobuaki Usui

(Fujitsu Lab.) who are coauthors of the author's papers on digital halftoning. He also would like to thank Hiro Ito (Kyoto Univ.), David Mount (Maryland Univ., U.S.A.), Masashi Kiyomi (Tokyo Univ.), and Koji Nakano (JAIST).

The work has been partially supported by the Scientific Grant-in-Aid by the Ministry of Education, Culture, Sports, Science and Technology of Japan and the Kayamori Foundation of Informational Science and Advancement.

References

1. T. Asano: "Digital Halftoning Algorithm Based on Random Space-Filling Curve," IEICE Trans. on Fundamentals, Vol.E82-A, No.3, pp.553-556, Medgeh 1999.
2. T. Asano, K. Obokata, N. Katoh, and T. Tokuyama: "Matrix rounding under the L_p-discrepancy measure and its application to digital halftoning," Proc. ACM-SIAM Symposium on Discrete Algorithms, pp.896-904, San Francisco, 2002.
3. T. Asano, T. Matsui, and T. Tokuyama: "Optimal Roundings of Sequences and Matrices," Nordic Journal of Computing, Vol.7, No.3, pp.241-256, Fall 2000.
4. T. Asano and T. Tokuyama: "How to Color a Checkerboard with a Given Distribution — Matrix Rounding Achieving Low 2×2- Discrepancy," Proc. ISAAC01, pp. 636-648, Christchurch, 2001.
5. T. Asano, D. Ranjan and T. Roos: "Digital halftoning algorithms based on optimization criteria and their experimental evaluation," IEICE Trans. Fundamentals, Vol. E79-A, No. 4, pp.524-532, April 1996.
6. B. E. Bayer: "An optimum method for two-level rendition of continuous-tone pictures," Conference Record, IEEE International Conference on Communications, 1, pp.(26-11)–(26-15), 1973.
7. B. Chazelle: "The Discrepancy Method: Randomness and Complexity," Cambridge University Press, 2000.
8. R. W. Floyd and L. Steinberg: "An adaptive algorithm for spatial gray scale," SID 75 Digest, Society for Information Display, pp.36–37, 1975.
9. D. E. Knuth: "Digital halftones by dot diffusion," ACM Trans. Graphics, 6-4, pp.245–273, 1987.
10. D.L. Lau and G.R. Arce: "Modern Digital Halftoning," Marcel Dekker, Inc., New York, 2001.
11. J. Matoušek: "Geometric Discrepancy," Springer, 1991.
12. T. Mitsa and K.J. Parker: "Digital halftoning technique using a blue-noise mask," J. Opt. Soc. Am., A/Vol.9, No.11, pp.1920-1929, 1992.
13. R. Morelli: "Pick's theorem and Todd class of a toric variety," Adv. Math., 100, pp.183-231, 1993.
14. R. Motwani and P. Raghavan: "Randomized Algorithms," Cambridge University Press, 1995.
15. V. Ostromoukhov, R.D. Hersh, and I. Amidror: "Rotated Dispersed Dither: a New Technique for Digital Halftoning," Proc. of SIGGRAPH '94, pp.123-130, 1994.
16. P. Raghavan and C.D. Thompson: "Randomized rounding," Combinatorica, 7, pp.365-374, 1987.
17. R. Ulichney: Digital halftoning, MIT Press, 1987.

18. R.A. Ulichney: "Dithering with blue noise," Proc. IEEE, 76, 1, pp.56-79, 1988.
19. R. Ulichney: "The void-and-cluster method for dither array generation," IS&T/SPIE Symposium on Electronic Imaging Science and Technology, Proceedings of Conf. Human Vision, Visual Processing and Digital Display IV, (Eds. Allebach, John Wiley), SPIE vol.1913, pp.332-343, 1993.