IEICE TRANS. FUNDAMENTALS, VOL.E90–A, NO.3 MARCH 2007

PAPER

A Pseudo-Hilbert Scan for Arbitrarily-Sized Arrays

Jian ZHANG[†a)], ***Student Member***, **Sei-ichiro KAMATA**[†b)], ***and*** **Yoshifumi UESHIGE**[††c)], ***Members***

SUMMARY The 2-dimensional (2-D) Hilbert curve is a one-to-one mapping between 2-D space and one-dimensional (1-D) space. It is studied actively in the area of digital image processing as a scan technique (Hilbert scan) because of its property of preserving the spacial relationship of the 2-D patterns. Currently there exist several Hilbert scan algorithms. However, these algorithms have two strict restrictions in implementation. First, recursive functions are used to generate a Hilbert curve, which makes the algorithms complex and computationally expensive. Second, both sides of the scanned rectangle must have same size and each size must be a power of two, which limits the application of the Hilbert scan greatly. In this paper, a Pseudo-Hilbert scan algorithm based on two look-up tables is proposed. The proposed method improves the Hilbert scan to be suitable for real-time processing and general application. The simulation indicates that the Pseudo-Hilbert scan can preserve point neighborhoods as much as possible and take advantage of the high correlation between neighboring lattice points. It also shows competitive performance of the Pseudo-Hilbert scan in comparison with other scan techniques.

key words: *space-filling curve, Hilbert curve, Euclidean distance, look-up tables, Pseudo-Hilbert scan*

1. Introduction

A space-filling curve is a one-to-one mapping between N-dimensional (N-D) space and 1-D space [14], [15]. By mapping each point in a multidimensional space into a 1-D space, the complex multidimensional access method can be transformed into a simple 1-D one. Hence a lot of research has been done and there exist many space-filling curves [8]. Among them, the Hilbert curve preserves point neighborhoods as much as possible [10], so it is applied widely in digital image processing, such as image compression [1], [9], [12], clustering an image [2], [13] and pattern recognition [11], [12], [16]. Currently there exist several algorithms [3]–[5] proposed for the 2-D Hilbert scan, such as the Kamata algorithm [3], [4], the Agui algorithm [5] and the Quinqueton algorithm [6]. However, these algorithms have more or less restrictions on their applications, and this makes them very difficult to be put in practice. For example, Agui and Quinqueton used the recursive functions to generate the scanning curve, and their algorithms were complex and took time to compute the one-to-one mapping correspondence. So the algorithms were very difficult to apply in real-time systems. In 1999 Kamata [4] proposed a fast method using look-up tables instead of the original recursive methods. However, the algorithm also had strict restrictions. It required the scanned region must be a square and the length of each side must equal the power of two. The application of the Kamata algorithm was limited. So these restrictions should be removed in order to improve the Hilbert scan for general application. This generalized Hilbert scan is called 2-D Pseudo-Hilbert scan.

In this paper, a non-recursive Pseudo-Hilbert scan algorithm based on the look-up tables [3], [4] is proposed for an arbitrarily-sized rectangle. It requires little memory for representing 2-D space data using the terminal and induction tables. And these two look-up tables are given a priori, this makes the algorithm keep the computational cost low. Therefore the proposed algorithm is suitable for real-time processing and easy to implement in hardware. Moreover, since the size of a scanned rectangle is arbitrary, the new scan technique has wider application than the original one. Note that although the scan is Pseudo-Hilbert, it holds the neighborhoods property as well. That is to say, the Pseudo-Hilbert scan can also preserve point neighborhoods as much as possible. This is demonstrated by the simulation at the end of the paper. The rest of paper is organized as follows. Section 2 gives the expression of the address assignment in the Hilbert scan. Then Sect. 3 presents a Pseudo-Hilbert scan algorithm for arbitrary-sized rectangles. In Sect. 4, we provide experimental evidence to demonstrate the property of preserving spacial locality for the Pseudo-Hilbert scan. Finally, in Sect. 5, we discuss the contributions of this paper and suggest future work.

2. A Hilbert Scan Curve and the Address Assignment

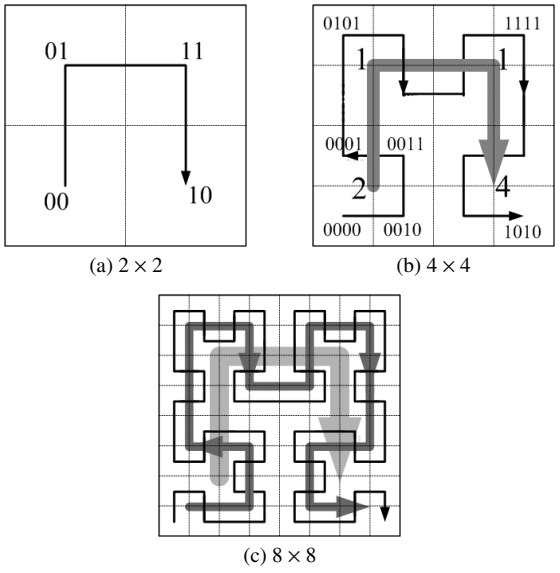
In 1891 Hilbert [7] drew a curve having the space-filling curve property in 2-D space. He demonstrated that the subsquares can be arranged so that the inclusion relationships are preserved, that is, if a square corresponds to an interval, then its subsquares correspond to subintervals of that interval. Figure 1 describes how this process is to be carried out and shows 2-D Hilbert curves with different resolutions. In the figure, the binary number expresses the address alignment which will be discussed in the following paragraph.

In this section, we define the expression of the points in

Manuscript received July 31, 2006.
Manuscript revised October 31, 2006.
Final manuscript received November 24, 2006.

[†]The authors are with the Graduate School of Information, Production and Systems, Waseda University, Kitakyushu-shi, 808-0135 Japan.
[††]The author is with Institute of System & Information Technologies/KYUSHU, Fukuoka-shi, 814-0001 Japan.
a) E-mail: zj_jay@toki.waseda.jp
b) E-mail: kam@waseda.jp
c) E-mail: ueshige@isit.or.jp

DOI: 10.1093/ietfec/e90–a.3.682



(a) 2×2 (b) 4×4

(c) 8×8

Fig. 1 2-D Hilbert scan.

a rectangle with the size $L_x \times L_y$, where L_x and L_y mean the length of horizontal and vertical side, respectively. In order to discuss expediently, we assume $L_x \geq L_y$ in this study. The coordinates of a point in 2-D space are denoted as (X, Y). The rectangle can be represented as a set of points

$$R(L_x, L_y) = \{(X, Y) | 0 \leq X < L_x, 0 \leq Y < L_y\}, \quad (1)$$

where X, Y, L_x and L_y are integer numbers.

Definition 2.1: *For any integer number l ($l \geq 2$), we can always represent it as $2 \cdot 2^m \leq l < 4 \cdot 2^m$. We denote m as the binary division times of l. And it can be calculated by the equation $\lfloor \log_2(\frac{l}{2}) \rfloor$, where the operator $\lfloor z \rfloor$ means the integer part of a real number z.*

Thus the binary division times of both sides of $R(L_x, L_y)$ can be obtained by

$$M_x = \left\lfloor \log_2 \left(\frac{L_x}{2} \right) \right\rfloor, \quad (2)$$

$$M_y = \left\lfloor \log_2 \left(\frac{L_y}{2} \right) \right\rfloor. \quad (3)$$

Because the previous assumption $L_x \geq L_y$, it is easy to get $M_x \geq M_y$. Thus, the address of a point in $R(L_x, L_y)$ can be expressed as a $2(M_x + 2)$-bit binary number,

$$\underbrace{\overbrace{x_{M_x+1}y_{M_x+1}}^{2bits}\overbrace{x_{M_x}y_{M_x}}^{2bits}\cdots\cdots\overbrace{x_0y_0}^{2bits}}_{2(M_x+2)bits}. \quad (4)$$

And the coordinates of a point (X, Y) can be expressed as two $(M_x + 2)$-bit binary numbers,

$$X = x_{M_x+1}x_{M_x}\cdots\cdots x_1x_0 \quad (5)$$

$$Y = y_{M_x+1}y_{M_x}\cdots\cdots y_1y_0 \quad (6)$$

where x_m and y_m ($0 \leq m \leq M_x + 1$) are 0 or 1. We can use these two binary sequences for representing the address of each point in a rectangle. In fact an $(M_y + 2)$-bit binary number is enough to express the address of Y because l_y has only M_y division times. We denote $\triangledown M = M_x - M_y$. So in the binary sequences $y_{M_x+1}y_{M_x}\cdots\cdots y_1y_0$, $\triangledown M$-bit binary number $y_{3+\triangledown M-2}y_{3+\triangledown M-3}\cdots y_2$ always equals to 0. And the memory can be saved effectively by eliminating these bits. However, for simplicity, we use an $(M_x + 2)$-bit binary number to represent address in this study.

3. A Pseudo-Hilbert Scan Algorithm

In this section, we generalize the 2-D Hilbert scan to be suitable for an arbitrary-sized rectangle $R(L_x, L_y)$, namely the Pseudo-Hilbert scan. The Pseudo-Hilbert scan algorithm includes two steps. At first, a division method is proposed to divide a rectangle into blocks. Then we summarize a computation method for the addresses in each block based on the look-up tables method [3], [4].

3.1 Division of a Rectangle

Since the size of a rectangle is arbitrary and not always the power of two, we make a rule to divide each side of the rectangle. This rule is represented as the following function $division(l, l_0, l_1)$, which expresses how to divide a side with the length l into two edges with the length: l_0 and l_1.

***division*(l, l_0, l_1)**

begin
$m = \left\lfloor \log_2 \left(\frac{l}{2} \right) \right\rfloor$;
If $2 \cdot 2^m \leq l < 3 \cdot 2^m$, *then* $l_0 \longleftarrow l - 2^m$ *and* $l_1 \longleftarrow 2^m$;
If $l = 3 \cdot 2^m$, *then* $l_0 \longleftarrow 2 \cdot 2^m$ *and* $l_1 \longleftarrow 2^m$;
If $3 \cdot 2^m < l < 4 \cdot 2^m$, *then* $l_0 \longleftarrow l - 2 \cdot 2^m$ *and* $l_1 \longleftarrow 2 \cdot 2^m$;
end

Perform above procedure repeatedly, the side can be divided into 2^m edges finally. The following program shows the case of the vertical side with the length L_y, where $l_p^{(q)}$ means the p-th edge after the q-th division.

0) *Initialize* $l_0^{(0)} \leftarrow L_y$;

1) $division(l_0^{(0)}, l_0^{(1)}, l_1^{(1)})$;

2) $division(l_0^{(1)}, l_0^{(2)}, l_1^{(2)})$ *and* $division(l_1^{(1)}, l_2^{(2)}, l_3^{(2)})$;
$\vdots \quad \vdots \quad \vdots$

k) $division(l_0^{(k-1)}, l_0^{(k)}, l_1^{(k)})$, $division(l_1^{(k-1)}, l_2^{(k)}, l_3^{(k)}), \cdots,$ $division(l_{2^{k-1}-1}^{(k-1)}, l_{2^k-2}^{(k)}, l_{2^k-1}^{(k)})$;
$\vdots \quad \vdots \quad \vdots$

M_y) $division(l_0^{(M_y-1)}, l_0^{(M_y)}, l_1^{(M_y)})$, $division(l_1^{(M_y-1)}, l_2^{(M_y)}, l_3^{(M_y)}), \cdots,$ $division(l_{2^{M_y-1}-1}^{(M_y-1)}, l_{2^{M_y}-2}^{(M_y)}, l_{2^{M_y}-1}^{(M_y)})$;

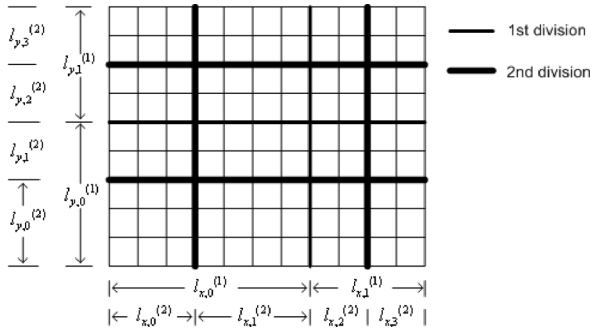


Fig. 2 Division process.

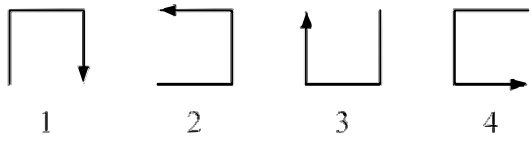


Fig. 3 Four basic patterns of 2-D Hilbert curves (the number 1–4 is curve type).

Similarly, the horizontal side of $R(L_x, L_y)$ can be divided M_x times. However, in order to make the scan comply to the Hilbert scan (mapping the square) in global sense, that is, the scanning of blocks submits to the Hilbert scan, we should divide the vertical side and horizontal side for the same times. So the number of its division times equals M_y ($M_y \leq M_x$). Figure 2 explains how to use the division method to split a rectangle $R(11, 9)$. After twice divisions ($M_y = 2$), each side is divided into four edges ($l^{(2)}_{x,0} \cdots l^{(2)}_{x,3}$ or $l^{(2)}_{y,0} \cdots l^{(2)}_{y,3}$). So the rectangle is divided into sixteen blocks.

3.2 The Hilbert Scan of Blocks

$R(L_x, L_y)$ can be divided into $2^{M_y} \times 2^{M_y}$ blocks based on the above discussion. So we can utilize the Hilbert curve to arrange these blocks. In this section, we introduce a fast Hilbert scan algorithm in a square $R(2^{M_y}, 2^{M_y})$. The algorithm traverses oct-tree with a depth of M_y using a depth first search. The scan from the k-th depth nodes to the $(k+1)$-th depth nodes is ordered by the look-up tables, where $k = 1, 2, \cdots, M_y$. We describe the look-up tables for the Hilbert scan in 2-D space in the following paragraph.

A Hilbert curve (parent region) $R(2^{M_y}, 2^{M_y})$ includes four subsquare regions (child region) which are congruent with $R(2^{M_y-1}, 2^{M_y-1})$. As shown in Fig. 3, we use four 2-D Hilbert curves to connect these four congruent subsquare regions. Each curve type is identified by a number from 1 to 4.

Figures 1(a) and (b) show addresses and curve types in four child regions, in the case where the curve type of a parent region is "1." When we arrange the four addresses and curve types in the order of scanning child regions, we obtain the address sequence as follows (see Fig. 1(a)):

$$00 \longrightarrow 01 \longrightarrow 11 \longrightarrow 10 \tag{7}$$

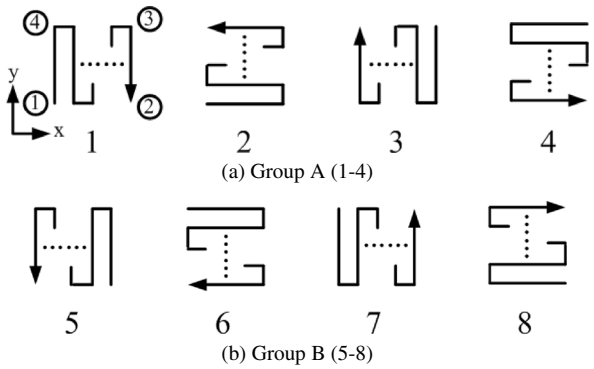


Fig. 4 Two group of scanning manners (the number 1–8 means the type).

and the curve type sequence as follows (see Fig. 1(b))

$$2 \longrightarrow 1 \longrightarrow 1 \longrightarrow 4. \tag{8}$$

To create these sequences, we prepare two look-up tables. One is a terminal table $\boldsymbol{T}_{trm}$ for the address sequences and the other is an induction table $\boldsymbol{T}_{ind}$ for the curve type sequences. We use two matrices — $T_{trm}[\gamma][i]$ and $T_{ind}[\gamma][i]$ — to represent the elements of the two tables, where $1 \leq \gamma \leq 4$ is the curve type of parent region and $1 \leq i \leq 4$ is an order in each sequence. Thus, they can be expressed as follows,

$$\boldsymbol{T}_{trm} = (T_{trm}[\gamma][i]) = \begin{pmatrix} 00 & 01 & 11 & 10 \\ 00 & 10 & 11 & 01 \\ 11 & 10 & 00 & 01 \\ 11 & 01 & 00 & 10 \end{pmatrix},$$

$$\boldsymbol{T}_{ind} = (T_{ind}[\gamma][i]) = \begin{pmatrix} 2 & 1 & 1 & 4 \\ 1 & 2 & 2 & 3 \\ 4 & 3 & 3 & 2 \\ 3 & 4 & 4 & 1 \end{pmatrix}.$$

Figure 1(a) is the case of $\gamma = 1$, the address is the first row of the terminal table $\boldsymbol{T}_{trm}$ and the order of the child region is the first row of the table $\boldsymbol{T}_{ind}$ shown in Fig. 1(b).

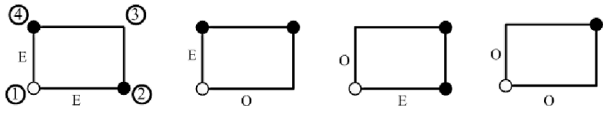
3.3 Address Assignment in Each Block

In the $2^{M_y} \times 2^{M_y}$ blocks, each corner point of any block has always two scanning manners,

1. Scanning along the x direction and then along the y direction;
2. It is the reverse of the first one, scanning along y direction then along x direction.

Figure 4 shows all the scanning manners of a block. In the figure, they are classified into two different groups (A and B) in accordance with the location of entry point. Group A includes the scanning manners starting from point ① or ③, and Group B includes the ones starting from ② or ④.

In a block, the length of each side is either an odd number or an even number. Thus the blocks are generally classified into four types which are shown in Fig. 5. In Fig. 5(b), $T_B(O, E)$ means a block has the odd and even length corresponding to the horizontal and vertical side respectively,



(a) $T_B(E,E)$ (b) $T_B(O,E)$ (c) $T_B(E,O)$ (d) $T_B(O,O)$

Fig. 5 The scanning manners of four block types, when the entry is point 1.

Table 1 Inclusion relationship between rectangle and block.

Rectangle type	Block type
$T_R(E,E)$	$T_B(E,E)$
$T_R(E,O)$	$T_B(E,O)$, $T_B(E,E)$
$T_R(O,E)$	$T_B(O,E)$, $T_B(E,E)$
$T_R(O,O)$	$T_B(O,O)$, $T_B(E,O)$, $T_B(O,E)$, $T_B(E,E)$

such as a block with the size 5×4. Utilizing the first and second scanning manners in these four block types, we can obtain the exit points of them. In each figure, the signs $\circ$ and $\bullet$ denote the entry and the exit respectively. In fact, Fig. 3 is a special case when Group A is used in the block with size 2×2.

Since L_x (or L_y) is an even number or odd number in $R(L_x, L_y)$, we can classify rectangles into four types — $T_R(E,E)$, $T_R(E,O)$, $T_R(O,E)$ and $T_R(O,O)$, where E or O represents the even length or odd length of a corresponding side (L_x or L_y). Then based the division method discussed in Subsection 3.1, it can be inferred that which block type should be included in each rectangle type. For example, a rectangle $R(7,5)$ which belongs to the type $T_R(O,O)$ includes four blocks with the size 3×3 ($T_B(O,O)$), 3×2 ($T_B(O,E)$), 4×3 ($T_B(E,O)$) and 4×2 ($T_B(E,E)$). Table 1 expresses this inclusion relationship. Therefore, we define the scanning manner in each block as follows in the light of the type of a rectangle (The validity is proved in appendix).

1. $T_R(E,E)$ rectangle region

   After M_y times division, a square $R(2^{M_y+1}, 2^{M_y+1})$ includes $2^{M_y} \times 2^{M_y}$ blocks, and the size of each block is 2×2. An appropriate pattern of Hilbert curve (Fig. 3) is selected to scan each block according to the table $\boldsymbol{T}_{trm}$. Now as to an arbitrary $T_R(E,E)$ rectangle, since the size of block is not always 2×2, we use four scanning manners of Group A shown in Fig. 4(a) instead of the basic patterns of Hilbert curve to scan each block. The scanning manner of each block is also decided by the table $\boldsymbol{T}_{trm}$.

2. $T_R(O,E)$ rectangle region

   For the first block ($T_B(O,E)$), we define that the scanning curve enters from point ① and exits from point ③ (the first scanning manner). For the other $T_B(O,E)$ blocks, entry is the point ② and exit is the point ③ (the eighth scanning manner). And the scanning manners of $T_B(E,E)$ blocks are selected from 5-8, which is decided by the entry point and the table $\boldsymbol{T}_{trm}$.

3. $T_R(E,O)$ rectangle region

   For the first block ($T_B(E,O)$), we also define that the scanning curve enters from point ① and exits from point ③ (the second scanning manner). For the scanning of other $T_B(E,O)$ blocks, entry is the point ④ and exit is the point ③ (the seventh scanning manner). And the scanning manners of $T_B(E,E)$ blocks are selected from 5-8, which is decided by the entry point and the table $\boldsymbol{T}_{trm}$.

4. $T_R(O,O)$ rectangle region

   The scanning manner of the first block ($T_B(O,O)$) starts from point ① and ends in point ③ (the first scanning manner). The scanning of $T_B(E,O)$ and $T_B(O,E)$ blocks is the same as that mentioned in item 2 and 3. The scanning manners of $T_B(E,E)$ blocks are also selected from 5-8, which are decided by the entry point and the table $\boldsymbol{T}_{trm}$.

In the 2nd, 3rd and 4th circumstances, because we define the scanning manner of the first block and the scanning order of blocks (the Hilbert scan), we always know the entry point of the current scanned block ($T_B(E,E)$) and the location (left, right, up or down) of the next block. Then we can decide the scanning manner of this $T_B(E,E)$ block. For example, assuming the entry is point ① shown in Fig. 5(a), we choose the first (second) scanning manner if the next block is located to the left or up (right or down).

In order to explain the algorithm, we give an example. Given a rectangle $R(5,5)$, it can be divided into four blocks according to the division rule method — $\{B(l_x,l_y) | (l_x,l_y) = (3,3),(3,2),(2,2),(2,3)\}$, where $B(l_x,l_y)$ represents a block with the size $l_x \times l_y$. Each block is identified by the number (from 1 to 4), which is the order of the block scan. Figure 6 shows the address for lattice point "P" which is located at $(X,Y) = (4,0)$. Since $M_y = 1$, the address of "P" needs $6(= 2M_y + 4)$-bit binary number to represent. We compute its address by the following steps:

1. $i(k)$, $\gamma(k)$ and $\alpha(k)$ mean a scanning order, a curve type and an address, respectively, where k means k-th splitting. $\gamma(0) = 1$ is given.
2. This step is the computation of the upper 2 bits in the address to "P." Both sides of $R(5,5)$ are divided into the same form — $3 + 2$. Thus, we obtained four blocks shown in Fig. 6. Figure 6 also shows that "P" is the shade region, the forth block in the order of the Hilbert scan. So we get $i(0) = 4$, and we know $\gamma(0) = 1$. Then, we obtain $\alpha_1 = T_{trm}[\gamma(0)][i(0)] = T_{trm}[1][4] = 10$.
3. This step is the computation of the lower 4 bits of the address. The block with $\alpha_1 = 10$ is congruent with $B(2,3)$ which belongs to the $T_B(E,O)$, and its entry point is $(3,2)$. The order of the point "P" is the forth one among the six lattice points in the forth block. Accordingly we obtained $\alpha_2 = 0100$.
4. Hence, the address of "P" is $\boldsymbol{\alpha} = \alpha_1\alpha_2 = 100100$.

To make a summary, the whole Pseudo-Hilbert scan

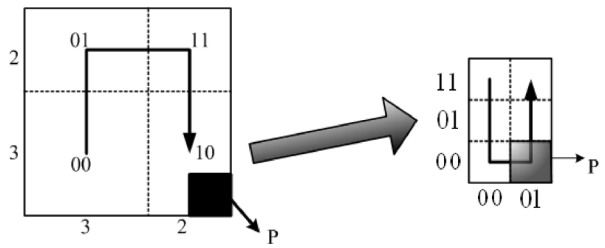


Fig. 6 An example of the address assignment to the lattice point "P" using Pseudo-Hilbert scan.

algorithm is shown as follows. It shows the Pseudo-Hilbert scan algorithm for generating all addresses ($(2M_x + 2)$-bit binary number) in a rectangle region $R(L_x, L_y)$. In initial condition, the value of $\gamma(0)$ equals one.

```
Pseudo-Hilbert scan
1. Address assignment of the blocks
for i_0 = 1, 2, ..., 4
  α_0 = x_{M_y-1} y_{M_y-1} = T_trm[γ(0)][i_0]
  γ(1) = T_ind[γ(0)][i_0]
        ⋮          ⋮
for i_m = 1, 2, ..., 4
    α_m = x_{M_y-m-1} y_{M_y-m-1} = T_trm[γ(m)][i_m]
    γ(m + 1) = T_ind[γ(m)][i_m]
          ⋮          ⋮
  for i_{M_y-1} = 1, 2, ..., 4
      α_{M_y-1} = x_0 y_0 = T_trm[γ(M_y − 1)][i_{M_y-1}]
2. Computation of address in each block
      l_x = l^{(M_y)}_{x_{M_y-1} x_{M_y-2} ... x_0}, l_y = l^{(M_y)}_{y_{M_y-1} y_{M_y-2} ... y_0}
      b = block_type(l_x, l_y)
      if b = T_B(O, E) or T_B(O, O)
        if it is the first block
          go to scanning_1
        else
          go to scanning_8
      else if b = T_B(E, O)
        if it is the first block
          go to scanning_2
        else
          go to scanning_7
      else
        s = scanning_type(α, γ(M_y − 1))
        go to scanning_s

      scanning_1:
        for i = 0, 1, ..., l_x − 1
          for j = 0, 1, ..., l_y − 1
          α_{M_y} = ij
          α = α_0 ... α_{M_y-1} α_{M_y}
          output < α >
      scanning_2:
        for i = 0, 1, ..., l_y − 1
          for j = 0, 1, ..., l_x − 1
          α_{M_y} = ji
          α = α_0 ... α_{M_y-1} α_{M_y}
          output < α >
            ⋮          ⋮
      scanning_8:
        for i = 0, 1, ..., l_y − 1
          for j = l_x − 1, ..., 1, 0
          α_{M_y} = ji
          α = α_0 ... α_{M_y-1} α_{M_y}
          output < α >
```

From the algorithm, it is easy to see that at the m-th splitting, we obtain a *two*-bit binary number α_m from the terminal table $\boldsymbol{T}_{trm}$ and the curve type $\gamma(m+1)$ from the induction table $\boldsymbol{T}_{ind}$. Here, $m(0 \le m \le M_y)$ is the number of splits. The procedures above are performed until $m = M_y$, and we obtain the upper $2M_y$ bits $(\alpha_0\alpha_1 \cdots \alpha_{M_y-1})$ in each address. In the following steps, we compute the lower $2(M_x - M_y + 2)$ bits α_{M_y}. Firstly, we use the function "*block_type*( )" to obtain the type of the scanned block $B(l_x, l_y)$. Then we select the scanning manner of the block based on the four circumstances mentioned above. The function "*scanning_type*( )" is used to make a choice from *scanning*_1 to *scanning*_8 when the scanned block is the type $T_B(E, E)$. Finally we can obtain the remnant binary number α_{M_y}.

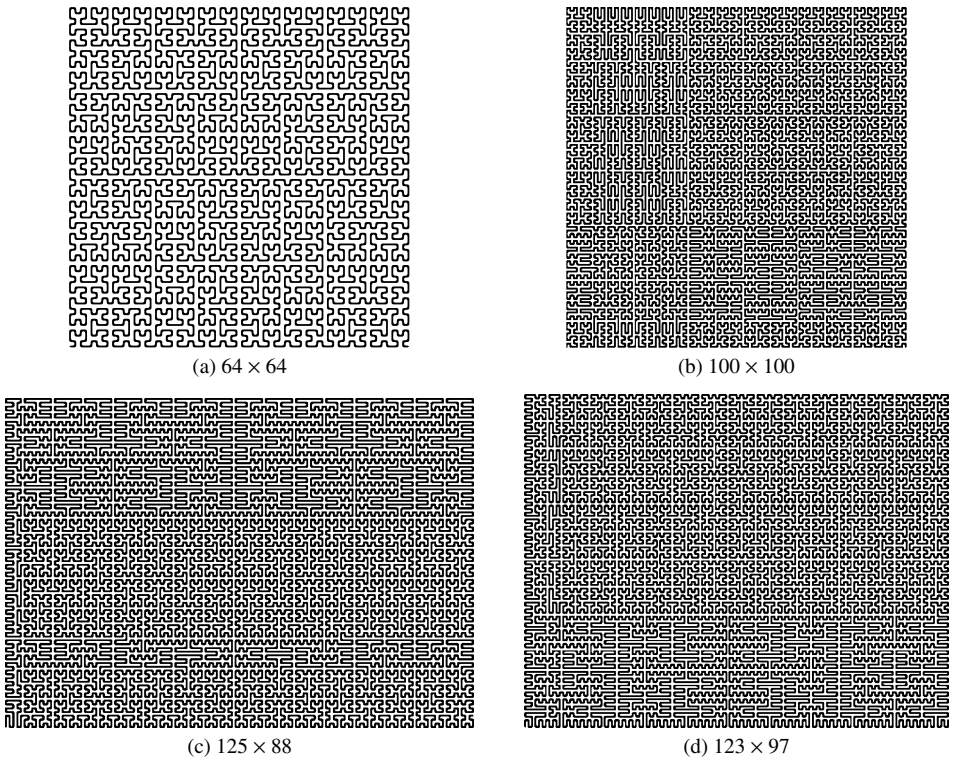
This algorithm is performed through referring to the look-up tables with lower complexity. Table 2 shows a comparison of storage memory and computation complexity between our method, the Kamata method and the recursive method. Only 64 bits are reserved for the look-up tables in our method, which is same as the Kamata method. While the recursive method requires $10M_y$-bits storage. When M_y increases, the recursive method requires much more storage memory. In the computation complexity, the recursive method takes considerably more time to compute all addresses. Although our method takes a little more expense than the Kamata method, this doesn't have any effect on its implementation in real-time processing and this cost is worthy for the general application of the Hilbert scan. Therefore, the proposed method is easy to implement and suitable for real-time systems.

Table 2 Comparison of computation complexity and storage memory.

Method	Computation complexity	Storage(bits)
Our method	$O(4^{M_y} l_x l_y)$	64
Kamata method	$O(2^{2M_y})$	64
Recursive method	$O(2^{4M_y + M_y 2^{M_y+1}})$	$10M_y$

4. Experimental Results

Four examples of the Pseudo-Hilbert scan are shown in Fig. 7. Figure 7(a) is the case of the Pseudo-Hilbert scan with the size 64×64. From this figure, it is easy to see that when the both sides of the region are power of two and equal, the structures and properties of the Pseudo-Hilbert scan are the same as that of the Hilbert scan. In other words,



(a) 64×64 (b) 100×100

(c) 125×88 (d) 123×97

Fig. 7 Examples of Pseudo-Hilbert scan for different sized rectangles.

the Pseudo-Hilbert scan is the Hilbert scan. Figures 7(b)–(d) show the cases of the Pseudo-Hilbert scan with the size 100×100, 88×125 and 97×123, respectively. In these figures, it is easy to find that the Pseudo-Hilbert scan submits to the Hilbert scan on the level of blocks, and the most parts of the Pseudo-Hilbert scan are same as the Hilbert scan. So we can conclude that the Pseudo-Hilbert scan has the similar property of the Hilbert scan preserving point neighborhoods as much as possible.

In order to demonstrate this property, we compared the proposed algorithm with some common scan methods using a statistical method [5], [10]. This statistical simulation has the following three steps,

1. Initialize two vectors $\boldsymbol{N} = 0$ and $\boldsymbol{L} = 0$;
2. Take two points $a(X_1, Y_1), b(X_2, Y_2)$ randomly in a rectangle region $R(L_x, L_y)$, where $1 \le X_1, X_2 \le L_x$ and $1 \le Y_1, Y_2 \le L_y$.
3. Then, calculate the square Euclidean distance d ($d \in [0, (L_x - 1)^2 + (L_y - 1)^2]$) and the scanning length l ($l \in [0, L_x \cdot L_y]$) between these two points.

$$d = (X_2 - X_1)^2 + (Y_2 - Y_1)^2; \tag{9}$$

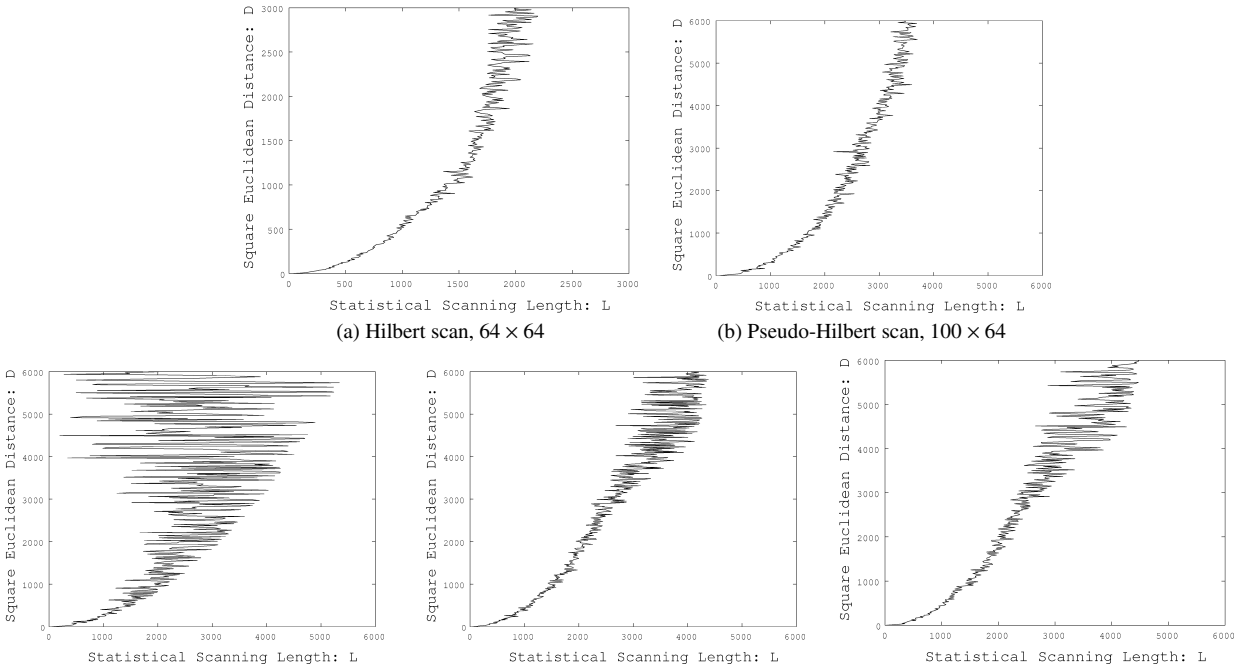
$$\boldsymbol{N}(d) = \boldsymbol{N}(d) + 1; \tag{10}$$

$$\boldsymbol{L}(d) = \boldsymbol{L}(d) + l. \tag{11}$$

Every iterating time $\boldsymbol{N}(d)$ stores the number of pairs of points whose square Euclidean distance equals d and $\boldsymbol{L}(d)$ stores the scanning length which is added to previous ones. After enough recursive trials (Step 2 and 3), we can compute the mean scanning length by the following equation.

$$\widetilde{\boldsymbol{L}}(d) = \boldsymbol{L}(d)/\boldsymbol{N}(d) \tag{12}$$

Figure 8 shows the relation between $\widetilde{\boldsymbol{L}}(d)$ and d. Figure 8(a) is the case of the Hilbert scan with the size 64×64. Figure 8(b) shows the one of the Pseudo-Hilbert scan with the size 100×64. Here, the correlation coefficient is used to evaluate similarity between these two scans, and it is equal to 0.9674. So the trends of both curves are similar, which illustrates that the Pseudo-Hilbert scan has similar properties as the Hilbert scan. In these two figures, in a range $[d, d + \Delta d]$ where Δd is small enough, the corresponding $\widetilde{\boldsymbol{L}}$ fluctuates in a very small scope. We can say that $\widetilde{\boldsymbol{L}}(d)$ and d are almost proportional, which proves that the Pseudo-Hilbert scan and Hilbert scan can preserve the high correlation of 2-D data. For a comparison the case of a Raster scan shown in Fig. 8(c), it is indicated that its trends is very different from the case of the Hilbert scan (the correlation coefficients is 0.6494) and $\widetilde{\boldsymbol{L}}$ fluctuates in a big range. For example, when d is about 2000, $\widetilde{\boldsymbol{L}}$ fluctuates from 1800 to 3400 in the case of the Raster scan, while it fluctuates in a much smaller scope from 2100 to 2300 in the case of the Pseudo-Hilbert scan. So the Pseudo-Hilbert scan has a much better proportional property between $\widetilde{\boldsymbol{L}}(d)$ and d. When d increases, this performance of the Raster scan deteriorates significantly, while the proposed method still holds the good performance. Figures 8(d) and (e) are the ones of two different kinds of mixing scans in a rectangle $R(100, 64)$. The combination of the Hilbert scan and the Raster scan is shown in Fig. 8(d). First the rectangle is divided into two parts — a subsquare of 64×64 and a subrectangle of 36×64, then



(a) Hilbert scan, 64×64 (b) Pseudo-Hilbert scan, 100×64

(c) Raster scan, 100×64 (d) Mixing of Hilbert scan and Raster scan, 100×64 (e) Mixing of Hilbert scan and Diagonal scan, 100×64

Fig. 8 Relation between square Euclidean distance d and the statistical scanning length $\widetilde{L}(d)$ between two points.

the Hilbert scan is used in the square and the Raster scan is used in the subrectangle. Figure 8(e) is the case that use the Diagonal scan instead of the Raster scan in the subrectangle. Comparing with the Raster scan, the performance is improved greatly in these two cases. However, they still don't solve the problem that the proportional property becomes bad intensely when d increases ($d \geq 3000$). The Pseudo-Hilbert scan shows its robustness and gives competitive performance in this aspect.

5. Conclusions

In this paper, a Pseudo-Hilbert scan algorithm is proposed for any arbitrarily-sized rectangle region. The proposed algorithm based on the look-up table method has low computational complexity and fast scan, which makes it suitable for real-time processing. Furthermore, our algorithm doesn't have any restriction on the size of the scanned rectangle. Thus the constraint of the Hilbert scan is significantly relaxed, which undoubtedly leads to many new applications in those areas can benefit from reducing the dimensionality of the problem, such as heuristics in computation (the traveling salesman problem). At the end of paper, we show the simulation. Through the results, it is demonstrated that the neighborhood property of the Hilbert scan is also preserved in the Pseudo-Hilbert scan. However, generalization of our algorithm for three and N-dimensional Pseudo-Hilbert scan is also faced with lots of problems, and how to overcome these problems will be our principle research in future.

Acknowledgments

This work was supported by fund from the Japanese MEXT (Ministry of Education, Culture, Sport, Science and Technology) via the Kitakyushu Innovative Cluster Project and Waseda University Grant for Special Research Projects (2005B-365).

References

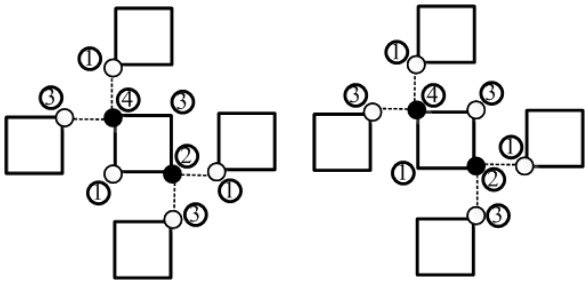
[1] S. Biswas, "Hilbert scan and image compression," Proc. IEEE Int. Conf. on Pattern Recognition, pp.201–210, 2000.
[2] B. Moon, H.V. Jagadish, and C. Faloutsos, "Analysis of the clustering properties of the Hilbert space-filling curve," IEEE Trans. Knowl. Data Eng., vol.13, no.1, pp.124–141, 2001.
[3] S. Kamata and Y. Bandoh, "An address generator of a Pseudo-Hilbert scan in a rectangle region," Proc. IEEE Int. Conf. on Image Process., pp.707–710, 1997.
[4] S. Kamata, R.O. Eason, and Y. Bandou, "A new algorithm for N-dimensional Hilbert scanning," IEEE Trans. Image Process., vol.8, no.7, pp.964–973, 1999.
[5] T. Agui, T. Nagae, and M. Nakajima, "Generalized Peano scans for arbitrary-sized arrays," IEICE Trans., vol.E74, no.5, pp.1337–1342, May 1991.
[6] J. Quinqueton and M. Berthod, "A locally adaptive Peano scanning algorithm," IEEE Trans. Pattern Anal. Mach. Intell., vol.3, no.4, pp.409–412, 1981.
[7] D. Hilbert, "Uber die stetige Abbildung einer Linie auf ein Flachenstuck," Mathematische Annalen, vol.38, pp.459–460, 1891.
[8] H. Sagan, Space-filling Curve, Springer Verlag, 1994.
[9] A. Lempel and J. Ziv, "Compression of two dimensional data," IEEE Trans. Inf. Theory, vol.IT-32, no.1, pp.2–8, 1986.
[10] C. Gotsman and M. Lindenbaum, "On the metric properties of

discrete space-filling curves," IEEE Trans. Image Process., vol.5, pp.794–797, 1996.
[11] K. Abend, T.J. Harley, and L.N. Kanal, "Classification of binary random patterns," IEEE Trans. Inf. Theory, vol.IT-11, no.4, pp.538–544, 1965.
[12] N. Memon, D.L. Neuhoff, and S. Shende, "An analysis of some common scanning techniques for lossless image coding," IEEE Trans. Image Process., vol.9, no.11, pp.1837–1848, 2000.
[13] H.V. Jagadish, "Linear clustering of objects with multiple atributes," Int. Conf. on Management of Data, pp.332–342, 1990.
[14] A.R. Butz, "Space filling curves and mathematical programming," Trans. Information and Control, vol.12, no.4, pp.314–330, 1968.
[15] A.R. Butz, "Convergence with Hilbert's space filling curve," J. Comput. Syst. Sci., vol.3, no.2, pp.128–146, 1969.
[16] L. Tian, S. Kamata, K. Tsuneyoshi, and H.J. Tang, "A fast and accurate algorithm for matching images using Hilbert scanning distance with threshold elimination function," IEICE Trans. Inf. & Syst., vol.E89-D, no.1, pp.290–297, Jan. 2006.

Appendix

Firstly, we consider the scanning manner Group A (1-4) in Fig. 4(a). Figures A· 1(a) and (b) show how the $T_B(E,E)$ block connects with four adjacent $T_B(E,E)$ blocks ($\circ$ denotes the entry and $\bullet$ denotes the exit). The dashed line between $\circ$ and $\bullet$ means that the two points are adjacent. From the figure, it is easy to see that for any $T_B(E,E)$ block in a scanned $T_R(E,E)$ rectangle region, the location of the entry point must be point ① or ③, and the location of the exit must be point ② or ④. While for the group B (5-8), we can obtain the opposite results in same analytic way — the entry locates in point ② or ④, the exit locates in point ① or ③. Thus both groups of scanning manners can complete the scanning of $T_R(E,E)$ independently, the difference between them is only the location of entry and exit.

According to the look-up table method discussed in Sect. 3, the scanning order of the first column blocks must belong to the $T_{trm}[1][:]$, $T_{trm}[2][:]$ and $T_{trm}[3][:]$, and the scanning order of the first row blocks must belong to the $T_{trm}[1][:]$, $T_{trm}[2][:]$ and $T_{trm}[4][:]$, where $T_{trm}[m][:]$ means the m-th row of $\boldsymbol{T}_{trm}$. Thus, for the first column blocks and the first row blocks, the scanning order is always from down to up or left to right. In group B, these blocks have fixed scanning manners, that is, entry is point ② and exit is point ③ for the 1st column blocks, and entry is point ④ and exit is point ③ for the 1st row blocks.

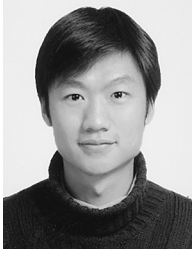


(a) The entry is point one (b) The entry is point three

Fig. A· 1 As to the scanning manners of Group A, the connection between adjacent blocks in a $T_R(E,E)$ rectangle.

Based on the discussion of Sect. 3, the $T_R(O,O)$ rectangle contains all the block types. And the $T_R(E,O)$ and $T_R(O,E)$ ones can be considered to be a specific form of $T_R(O,O)$. Therefore, if only the Pseudo-Hilbert scan is right for the $T_R(O,O)$ rectangle, it is correct for other three types of rectangles. So here we only prove the case of $T_R(O,O)$.

Firstly, according to the division method, we know all the $T_B(O,E)$ and $T_B(E,O)$ blocks allocate in the first column blocks and the first row blocks, respectively. Secondly, according to the scanning order of the first column and row blocks discussed above, we know that the scanning order of the $T_B(O,E)$ and $T_B(E,O)$ blocks are always from down to up or left to right. Thirdly, in the Fig. 5(d), we defined the entry of the first block is point ①, and exit is point ③. The entry of the next block (the right adjacent block or the up adjacent block) is point ② or ④, that is, the entry of the $T_B(O,E)$ block is point ② and that of the $T_B(E,O)$ block is point ④. According to the scanning manner 7 and 8, we choose the exit point of the $T_B(E,O)$ and $T_B(O,E)$ blocks as point ③. Thus, the effect of the $T_B(E,O)$ and $T_B(O,E)$ blocks in the scanning is same as that of the $T_B(E,E)$ blocks in Group B. So the $T_R(O,O)$ rectangle region can be scanned continuously and the curve is also the one-to-one mapping.



Jian Zhang received the B.E. in electrical engineering from Nanjing University of Information Science and Technology, Nanjing, China, in 2003, the M.E. degree in communication and information Engineering from Shanghai University, Shanghai, China, in 2005, and the second M.E. degree in computer science from Waseda University, Fukuoka, Japan, in 2006. He is now pursuing the Ph.D. degree at the Graduate School of Information, Production and Systems, Waseda University, Fukuoka, Japan. His current research is main on the space-filling curve application and image processing.

Sei-ichiro Kamata received the M.S. degree in computer science from Kyushu University, Fukuoka, Japan, in 1985, and the Doctor of Computer Science, Kyushu Institute of Technology, Kitakyushu, Japan, in 1995. From 1985 to 1988, he was with NEC, Ltd., Kawasaki, Japan. In 1988, he joined the faculty at Kyushu Institute of Technology. From 1996 to 2001, he has been an Associate Professor in the Department of Intelligent System, Graduate School of Information Science and Electrical Engineering, Kyushu University. Since 2003, he has been a professor in the Graduate School of Information, Production and Systems, Waseda University. In 1990 and 1994, he was a Visiting Researcher at the University of Maine, Orono. His research interests include image processing, pattern recognition, image compression, and space-filling curve application. Dr. Kamata is a member of the IEEE and the ITE in Japan.

Yoshifumi Ueshige received the B.E., M.E., and D.E. degrees from Kyushu Institute of Technology in 1992, 1994 and 1998, respectively. From 1997 to 2002 he worked at Kagoshima National College of Technology. In 2003 he was invited as a researcher at Kitakyushu Foundation for the Advancement of Industry, Science and Technology. From 2004 he works at Institute of Systems & Information Technologies/KYUSHU as a researcher. His research field includes image processing, and secure online authentication. He is a member of ITE, IPSJ, ACM, and IEEE.