PERGAMON

Pattern Recognition 34 (2001) 343–349

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Decoding structured light patterns for three-dimensional imaging systems

Yi-Chih Hsieh*

Department of Industrial Engineering, National Huwei Institute of Technology, 64 Wen-Hua Road, Huwei, Yunlin 632, Taiwan

Received 18 February 1999; received in revised form 8 August 1999; accepted 18 October 1999

Abstract

Structured light patterns may be used for acquiring 3-D range data with the use of a single camera. In such applications, the decoding problem, i.e., resolving the position of a particular "word" within a given structured light pattern, is of great importance. This paper presents a simple decoding algorithm for solving this problem. As shown, this proposed approach is better than "brute force" method. © 2000 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Structured light pattern; Decoding; Three-dimensional imaging; De Bruijn sequence

1. Introduction

Three-dimensional (3-D) range data are spatial coordinates for surface points of an object and they are useful for 3-D object matching, object recognition, and dimensional measurement [1]. A structured lighting system is a general method of acquiring surface range data from a scene. For such systems, an object is illuminated from a structured light source and this scene is imaged by a camera. The structured lighting system for 3-D imaging system is shown in Fig. 1.

For the specific point of the object in Fig. 1, if (u, v) is the corresponding coordinate detected by the camera and (i, j) is the corresponding coordinate projected from the structured light, then the object range point (x, y, z) can be determined by solving the following linear equations:

$$E(x, y, z, 1)^{\mathrm{T}} = (0, 0, 0, 0)^{\mathrm{T}} \tag{1}$$

where

$$E=\begin{pmatrix} t_{11}-t_{13}u & t_{21}-t_{23}u & t_{31}-t_{33}u & t_{41}-t_{43}u \\ t_{12}-t_{13}v & t_{22}-t_{23}v & t_{32}-t_{33}v & t_{42}-t_{43}v \\ q_{11}-q_{13}i & q_{21}-q_{23}i & q_{31}-q_{33}i & q_{41}-q_{43}i \\ q_{12}-q_{13}j & q_{22}-q_{23}j & q_{32}-q_{33}j & q_{41}-q_{43}j \end{pmatrix},$$

$T = [t_{ij}]$ is the transformation between the object and the image of the camera, and $Q = [q_{ij}]$ is the transformation between the object and the point of structured light. The values for T and Q can be found by using a calibration object [2].

Vuylsteke and Oosterlinck [3] introduced a special binary-encoded light pattern that may be used for a structured lighting system to acquire 3-D range data with the use of a single camera. Since only a single camera is required, unlike the approaches of Altschuler et al. [2] and Posdamer and Altschuler [4], their approach can be used in a dynamic environment (i.e., a scene with moving objects) for 3-D imaging systems. However, the proposed binary-encoded pattern has the disadvantage of not guaranteeing unique correspondence between projected beams and imaged beams, and is also subject to occlusion problems [5]. Subsequently, to overcome the problem, Griffin et al. [5] introduced a special encoded

* Tel.: + 886-5-632-9643; fax: + 886-5-6311548.
E-mail address: yhsieh@sparc.nhit.edu.tw (Y.-C. Hsieh).

PII: S0031-3203(99)00224-1

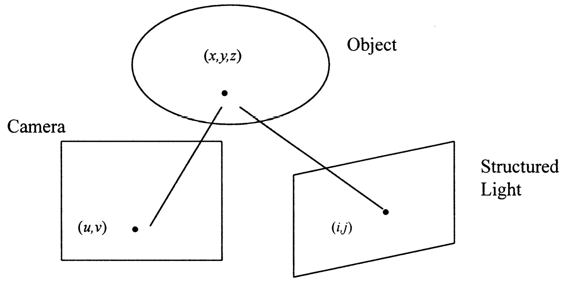

Fig. 1. Structured lighting system.

light pattern which may be used for a structured lighting system to acquire 3-D range data. Their encoded light pattern guarantees the unique correspondence between projected beams and imaged beams. The main idea of their structured light pattern is that: such a pattern may be projected onto a scene, and then a single camera is used to image the projected light beams, or so-called "words". Since any possible "word" within a specified structured light pattern is unique, the correspondence between the projected beams from the pattern and imaged beams is easily determined. We refer the interested readers to Griffin et al. [5] and Yee and Griffin [1] for details of the use of the structured light pattern.

Recently, Hsieh [6] pointed out that the sequences used to construct the structured light pattern of Griffin et al. [5] are indeed special cases of de Bruijn sequences (dBS) and proposed a simple algorithm for the construction of various de Bruijn sequences. Although the construction methods for structured light patterns have been proposed by several authors, e.g., Griffin et al. [5], Yee and Griffin [1], and Hsieh [6], the problem of decoding a given "word" within a specified structured light pattern has virtually ignored, even though its solution is important for the applications of acquiring 3-D range data. By a decoding algorithm we mean an algorithm for computing the position of a given "word" within a specified structured light pattern. As noted, the "brute force" method may be used for the decoding of structured light patterns by storing a complete look-up table of words and their positions. However, such a method quickly becomes infeasible because of the storage requirements as the structured light pattern increases in size.

The main purpose of this paper is to propose a simple algorithm for solving the decoding problem for structured light pattern in 3-D imaging systems. As shown, this proposed approach is better than "brute force" method. This research may provide a useful reference for researchers and practitioners attempting to decode the structured light pattern in acquiring 3-D range data. The paper is organized as follows. In the next section, we introduce the construction method for dBS and present two new simple decoding algorithms for the generated dBS. Section 3 describes an efficient decoding algorithm for structured light patterns. Finally, Section 4 offers brief concluding comments.

2. The generation and decoding of de Bruijn sequence

2.1. The generation of (m,n) de Bruijn sequence

Given m and n, finding an m-ary span n dBS, or simply an (m,n) dBS, is equivalent to finding an Eulerian circuit in its corresponding digraph $D_{m,n}$, in which the vertex set of $D_{m,n}$ is the set of all distinct m^{n-1} subsequences of length $n-1$ over the symbol set $\{1,2,\ldots,m\}$ [7]. Based upon the set of vertices and the set of arcs, one may represent the de Bruijn digraph $D_{m,n}$ by an $m^{n-1}\times m^{n-1}$ square adjacency matrix A [8], where

$$A_{ij}=\begin{cases}1 & \text{if } 1\leqslant j-[(i-1)m+1 \bmod m^{n-1}]-1\leqslant m\\ 0 & \text{otherwise}\end{cases}$$

$$\text{for } i,j\in\{1,2,\ldots,m^{n-1}\}.$$

Thus, finding an Eulerian circuit in the de Bruijn digraph $D_{m,n}$ is equivalent to assigning numbers of $1,2,3,\ldots,m^n$, in a particular way, to m^n cells $(i,j)\in T\equiv\{(i,j)|A_{ij}=1\}$ of the adjacency matrix A, where $1,2,3,\ldots,m^n$ indicating the ordering of arcs to be traveled in the de Bruijn digraph $D_{m,n}$. In general, if t is assigned to cell (i,j), then the next positive integer $t+1$ must be assigned to an unassigned cell of row j. The value of arc (i,j), from the ith vertex to the jth vertex in $D_{m,n}$, is defined as: $r_{ij}\equiv(j \bmod m)$. Note that throughout the paper we define $(m \bmod m)=m$ for $m\geqslant 2$.

Example 1. For $(m,n)=(4,2)$, the de Bruijn digraph $D_{4,2}$ is shown in Fig. 2. Its corresponding adjacency matrix A is shown in Fig. 3. Fig. 4 illustrates a possible assignment (Euler circuit) of (4,2) de Bruijn digraph $D_{4,2}$. Following the ordering of arcs (assignments) in Fig. 4 and appending their corresponding arc values r_{ij}, a (4,2) dBS is given by 44342 41332 31221 1. A general generation algorithm for (m,n) dBS as below [6]. As shown, the algorithm has several advantages over typical approaches, including Fredricksen and Kessler [9], Fredricksen and Maiorana [10], and Ralston [11].

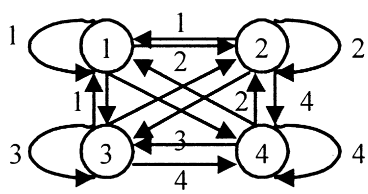

Fig. 2. The de Bruijn digraph for $D_{4,2}$.

$$\begin{array}{ccc|cccc|}
\text{arc value } r_{ij} & \Rightarrow & & 1 & 2 & 3 & 4 \\
\text{vertex} & \Rightarrow & & 1 & 2 & 3 & 4 \\
\Downarrow & & i\backslash j & 1 & 2 & 3 & 4 \\
\hline
1 & & 1 & 1 & 1 & 1 & 1 \\
2 & & 2 & 1 & 1 & 1 & 1 \\
3 & & 3 & 1 & 1 & 1 & 1 \\
4 & & 4 & 1 & 1 & 1 & 1 \\
\hline
\end{array}$$

Fig. 3. The adjacency matrix A for the de Bruijn digraph $D_{4,2}$.

$$\begin{array}{ccc|cccc|}
\text{arc value } r_{ij} & \Rightarrow & & 1 & 2 & 3 & 4 \\
\text{vertex} & \Rightarrow & & 1 & 2 & 3 & 4 \\
\Downarrow & & i\backslash j & 1 & 2 & 3 & 4 \\
\hline
1 & & 1 & 16 & 13 & 8 & 1 \\
2 & & 2 & 15 & 14 & 11 & 6 \\
3 & & 3 & 12 & 10 & 9 & 4 \\
4 & & 4 & 7 & 5 & 3 & 2 \\
\hline
\end{array}$$

Fig. 4. One possible assignment of Eulerian circuit for the de Bruijn digraph $D_{4,2}$.

Algorithm $G(m,n)$ (Constructing (m,n) dBS)
(Input: m and n; Output: S, an (m,n) dBS)

0. $i \leftarrow k \leftarrow 1$, $I \leftarrow (m,m,\cdots,m) \in R^{m^{n-1}}$, S is an empty sequence and T is an $m^{n-1} \times m$ empty matrix.
1. While $L(S) < m^n$ do
   begin
   $j \leftarrow I[i] + [(i-1)m + 1 \bmod m^{n-1}] - 1$;
   $k_2 \leftarrow j \bmod m$. Append k_2 to S.
   $T[i,k_2] \leftarrow k$;
   $I[i] \leftarrow I[i] - 1$;
   $k \leftarrow k + 1$;
   $i \leftarrow j$.
   end (while)

For example, $G(4{,}2) = 44342\ 41332\ 31221\ 1$ and $G(4{,}3) = 44434\ 42441\ 43343\ 24314\ 23422\ 42141\ 34124\ 11333\ 23313\ 22321\ 31231\ 12221\ 2111$. Note that $L(S)$ is the length of sequence S, and T is used for the decoding of sequence S in the following.

2.2. The decoding of (m,n) de Bruijn sequence

Method 1. Suppose that an (m,n) dBS S is constructed by Algorithm G, then the decoding for a given n-tuple subsequence of S is not difficult. As an example, consider the sequence $G(4{,}2)$. Its corresponding matrix T is exactly the same as that of Fig. 4. Thus the position for the 2-tuple subsequence 41 is $T[4{,}1] - 1 = 7 - 1 = 6$, i.e., 41 is located in the 6th position of $G(4{,}2)$. In general, the position of n-tuple subsequence $s_1 s_2 \ldots s_n$ in $G(m,n)$ is given by

$$T\left[\sum_{i=1}^{n-1} m^{n-i-1} s_i - \sum_{i=1}^{n-2} m^i, s_n\right] - n + 1.$$

The proof is straightforward and omitted since the matrix T is used to store the positions of dBS.

For an (m,n) dBS, the direct look-up table for the "brute force" method contains m^n subsequences. Since each subsequence has n-tuple, the total space required for the brute force decoding algorithm is proximately nm^n. However, this proposed decoding method requires only m^n total space.

Method 2. Next we present more efficient decoding algorithms for (m,n) dBS when $n = 2$ and 3, respectively. Before introducing the new decoding algorithm, we first note that the sequences constructed by Algorithm G are exactly the same as those of Fredricksen [12] by concatenating the so-called aperiodic necklaces. For example, for $(m,n) = (4{,}3)$, the lexicographic list of aperiodic necklaces is given by:

4(44)	3(33)	2(22)	1(11)
443	332	221	
442	331	211	
441	322		
433	321		
432	312		
431	311		
423			
422			
421			
413			
412			
411			

Thus the (4,3) dBS generated by Fredricksen's algorithm is 4 443 442 441 433 432 431 423 422 421 413 412 411 3 332 331 322 321 312 311 2 221 211 1 which is exactly the same as $G(4{,}3)$ by Algorithm G. The reason is that the proposed Algorithm G travels all vertices by the "right-vertex-first" priority (see Fig. 4). Thus decoding an (m,n) dBS with less storage space than m^n is possible. Next, we give the main decoding results for the cases of $n = 2$ and 3, respectively.

Lemma 1. *Let $s_i s_j$ be a 2-tuple aperiodic subsequence of $G(m,2)$ with $s_i \geqslant s_j$. Then the position of $s_i s_j$ in $G(m,2)$ is given by*

$$x(s_i, s_j) = m^2 - s_i^2 + \{2(s_i - s_j) - 1\}(1 - \delta_{ij}) + 1,$$

where δ_{ij} is a Kronecker delta, i.e., $\delta_{ij} = 1$ if $i = j$, otherwise, $\delta_{ij} = 0$.

Proof. See Appendix A.

Lemma 2. *Let $s_i s_j s_k$ be a 3-tuple aperiodic subsequence of $G(m, 3)$ with either ($s_i \geqslant s_j$ and $s_i > s_k$) or ($s_i = s_j = s_k$). Then the position of $s_i s_j s_k$ in $G(m, 3)$ is given by*

$$y(s_i, s_j, s_k) = \frac{m(m+1)(2m+1) - s_i(s_i+1)(2s_i+1)}{2} - \frac{(m-s_i)(3m+3s_i+1)}{2} + 3(s_i - s_j)(s_i - 1) + [3(s_i - s_k - 1) + 1] \times (1 - \delta_{ij}\delta_{jk}) + 1,$$

where δ_{ij} is a Kronecker delta.

Proof. See Appendix B.

Example 2. Consider $G(4,2)$. The position of 2-tuple subsequence 43 is $x(4,3) = m^2 - s_i^2 + \{2(s_i - s_j) - 1\}(1 - \delta_{ij}) + 1 = 16 - 16 + \{2(4-3) - 1\}(1-0) + 1 = 2$. Consider $G(4,3)$, the position of 3-tuple subsequence 423 is $y(4,2,3) = \{4(5)(9) - 4(5)(9)\}(1/2) - (4-4)(12+12+1)(1/2) + 3(4-2)(4-1) + \{3(4-3-1)+1\}(1-0) + 1 = 20$.

Based on the two Lemmas, we may decode any $G(m, 2)$ and $G(m, 3)$ for $m \geqslant 2$. The main idea is that any 2-tuple/3-tuple subsequence is near to a lexicographic necklace within 2/3 steps for $G(m, 2)/G(m, 3)$, respectively. As an example, consider the position of 212 in $G(4, 3)$. We first note that 212 is not in the list of aperiodic necklaces and we cannot directly use Lemma 2 for the position. However, the position of 221 is 58 (by Lemma 2) with one-step previous 212 in $G(4,3)$. Thus the position of 212 is $58 + 1 = 59$. Based on this idea, we give the following two general decoding algorithms for $G(m, 2)$ and $G(m, 3)$.

Algorithm $D_2(s_i, s_j)$ (Decoding $G(m, 2)$)
(Input: (s_i, s_j), a 2-tuple subsequence of $G(m, 2)$; Output: the position of $s_i s_j$)
begin
IF $s_i \geqslant s_j$, THEN compute $x(s_i, s_j)$ by Lemma 1, return x and stop;
ELSE IF $s_i > 1$, THEN compute $x(s_j, s_i)$, return $x + 1$ and stop;
ELSE compute $x(1 + s_j, s_i)$, return $x + 1$ and stop.
end

Example 3. For the $G(4,2)$, the position of 2-tuple subsequence 34 is $D_2(3,4) = x(4,3) + 1 = 2 + 1 = 3$ by the algorithm (note that $x(4,3) = 2$ by Example 2).

Algorithm $D_3(s_i, s_j, s_k)$ (Decoding $G(m, 3)$)
(Input: (s_i, s_j, s_k), a subsequence of $G(m, 3)$; Output: the position of $s_i s_j s_k$)
begin
1. IF $s_i = 1$ and $s_j = s_k = m$, THEN return $y = m^3$ and stop.
   IF $s_i = s_j = 1$ and $s_k = m$, THEN return $y = m^3 - 1$ and stop.
   IF ($s_i = s_j = s_k$) or ($s_i \geqslant s_j$ and $s_i > s_k$), THEN compute $y(s_i, s_j, s_k)$ by Lemma 2, return y and stop.
2. IF $s_j \geqslant s_k$ and $s_j > s_i$, THEN
   ELSE IF $s_i = 1$, THEN
   ELSE IF $s_j = s_k$, THEN compute $y(s_j + 1, 1, 1)$ by Lemma 2, return $y + 2$ and stop;
   ELSE compute $y(s_j, s_k + 1, 1)$ by Lemma 2, return $y + 2$ and stop;
   ELSE compute $y(s_j, s_k, s_i)$ by Lemma 2, return $y + 2$ and stop.
3. IF $s_k \geqslant s_i$ and $s_k > s_j$, THEN
   Else IF $s_i = s_j = 1$, THEN compute $y(s_k + 1, 1, 1)$ by Lemma 2, return $y + 1$ and stop;
   Else compute $y(s_k, s_i, s_j)$ by Lemma 2, return $y + 1$ and stop.
end

Example 4. For the $G(4, 3)$, the position of 3-tuple subsequence 342 is $D_3(3,4,2) = y(4,2,3) + 2 = 20 + 2 = 22$ by the algorithm (note that $y(4,2,3) = 20$ by Example 2).

3. The decoding of structured light patterns

Let $S_1 = G(m, 3)$ and $S_2 = G(m, 2)$. Griffin et al. [5] and Yee and Griffin [1] introduced the following procedure for the construction of structured light patterns.

Procedure (Constructing a structured light pattern)
(Input: S_1 and S_2; Output: W = a structured light pattern with size $m^2 \times m^3$)
begin
1. The first row of W is S_1, i.e., $W_{1,j} = S_1[j]$, $1 \leqslant j \leqslant m^3$;
2. For $2 \leqslant i \leqslant m^2$ and $1 \leqslant j \leqslant m^3$, $W_{i,j} = \{(W_{i-1,j} + S_2[i-1]) \bmod m\}$.
end

A structured light pattern constructed by $S_1 = G(4,3)$ and $S_2 = G(4,2)$ is shown in Fig. 5 [6].

Let $w = (w_{i-1,j}, w_{i,j-1}, w_{i,j}, w_{i,j+1}, w_{i+1,j})$ be a word as defined by Griffin et al. [5] and Yee and Griffin [1]. Next we present a new efficient decoding algorithm for computing the position of a given 5-tuple word w in the $m^2 \times m^3$ structured light pattern for $m \geqslant 3$.

```
4443442441433432431423422421413412411333233132232131231122212111
4443442441433432431423422421413412411333233132232131231122212111
4443442441433432431423422421413412411333233132232131231122212111
3332331334322321324312311314342341344222122421121424124411141444
3332331334322321324312311314342341344222122421121424124411141444
1114113112144143142134133132124123122444344243343242342233323222
1114113112144143142134133132124123122444344243343242342233323222
2221224223211214213241244243231234233111411314414313413344434333
1114113112144143142134133132124123122444344243343242342233323222
4443442441433432431423422421413412411333233132232131231122212111
2221224223211214213241244243231234233111411314414313413344434333
1114113112144143142134133132124123122444344243343242342233323222
2221224223211214213241244243231234233111411314414313413344434333
4443442441433432431423422421413412411333233132232131231122212111
2221224223211214213241244243231234233111411314414313413344434333
3332331334322321324312311314342341344222122421121424124411141444
```

Fig. 5. A 4-ary $4^2 \times 4^3$ structured light pattern [6].

Procedure (Decoding a structured light pattern)
(Input: a structured light pattern W and a given word w; Output: the position of the word w in W).
begin

1. Solve $(s_1 + w_{i-1,j}) = w_{i,j} (\bmod m)$ and $(s_2 + w_{i,j}) = w_{i+1,j} (\bmod m)$ for s_1 and s_2, where $s_1, s_2 \in \{1, 2, \ldots, m\}$;
2. By Algorithm D_2, let $x =$ the position of $s_1 s_2$ in S_2;
3. Solve $(t_k + w_{x,1}) = w_{i,j-2+k} (\bmod m)$ for t_k, $k = 1, 2, 3$, where $t_1, t_2, t_3 \in \{1, 2, \ldots, m\}$;
4. By Algorithm D_3, let $y =$ the position of $t_1 t_2 t_3$ in S_1;
5. Return $(x + 1, y + 1)$.

end

At Step 3, $w_{x,1}$ can be computed by the recursion $w_{i+1,1} = \{(w_{i,1} + S_2[i]) \bmod m\}$, where $w_{1,1} \equiv m$ and $1 \leqslant i \leqslant m^2$.

Example 5. Consider the word (4,2,3,1,3) of the structured light pattern in Fig. 5. Following the above procedure, at step 1 we solve $(s_1 + 4) = 3 (\bmod 4)$ and $(s_2 + 3) = 3 (\bmod 4)$, and have $s_1 s_2 = 34$. At Step 2, we have $D_2(3,4) = x(4,3) + 1 = 3$ (by Example 3). At Step 3, following the recursion we have $w_{4,1} = 3$. After solving $3 + t_1 = 2 (\bmod 4)$, $3 + t_2 = 3 (\bmod 4)$, and $3 + t_3 = 1 (\bmod 4)$, we have $t_1 t_2 t_3 = 342$. At Step 4, we obtain $D_3(3,4,2) = y(4,2,3) + 2 = 22$ (by Example 4). Thus, the position of word (4,2,3,1,3) in the structured light pattern of Fig. 5 is $(x + 1, y + 1) = (4,23)$.

For an $m^2 \times m^3$ structured light pattern, the direct look-up table for the "brute force" method requires $m^2 \times m^3$ entries for words, each has 5 tuples. Thus, the total space required for the brute force decoding algorithm is proximately $5 \times m^2 \times m^3$. It is clear that the proposed procedure requires less storage space.

4. Conclusions

Structured light patterns have been used for the acquiring of 3-D range data with the use of a single camera. Although the construction methods for structured light patterns have been proposed by several authors, the problem of decoding for a specified structured light pattern has virtually ignored, even though its solution is important for the applications of acquiring 3-D range data. The "brute force" method may be used for the decoding of structured light pattern by storing a complete look-up table of words and their positions. However, such a method quickly becomes infeasible for the storage requirements as the structured light pattern increases in size.

In this paper:

1. we have presented new and simple decoding algorithms for (m, 2) dBS and (m, 3) dBS. As shown these algorithms are all better than the brute force method in the storage requirements;
2. we have proposed a new and efficient algorithm for decoding the structured light patterns. As shown this approach is much better than the brute force method in the storage requirements.

This research may provide a useful reference for researchers and practitioners attempting to decode the structured light pattern in acquiring 3-D range data.

Acknowledgements

The author wishes to thank anonymous reviewers for helpful comments of this paper. This research is supported by National Science Council, Taiwan, under grant NSC 88-2213-E-150-002.

Appendix A

Proof of Lemma 1. For an $(m, 2)$ dBS, the lexicographic list of aperiodic necklaces is given by

Group	m	…	p	…	4	3	2	1
	$m(m)$	…	$p(p)$	…	4(44)	3(3)	2(2)	1(1)
	$m, m-1$		$p, p-1$		43	32	21	
	$m, m-2$	…	$p, p-2$	…	42	31		
	:		:		41			
	$m,1$		$p,1$					

Let $F(p)$ be the number of elements in group p, $p = 1, 2, 3, \ldots, m$. Thus, we have $F(p) = 1 + 2(p-1) = 2p - 1$.

Since $s_i \geqslant s_j$, there are two possible cases for s_i and s_j (Note that if $s_j > s_i$, then $s_i s_j$ is not in the lexicographic list of aperiodic necklaces).

1. If $s_i = s_j$, then the number of elements positioned before $s_i s_j$ is given by

$$\sum_{k=s_i+1}^{m} F(k) = \sum_{k=s_i+1}^{m} (2k-1) = m^2 - s_i^2.$$

   Thus the position of $s_i s_j$ in $G(m, 2)$ is $m^2 - s_i^2 + 1$.
2. If $s_i > s_j$, then the number of elements positioned before $s_i s_j$ is given by

$$\left(\sum_{k=s_i+1}^{m} F(k)\right) + 2(s_i - s_j) - 1$$

$$= m^2 - s_i^2 + 2(s_i - s_j) - 1.$$

   Thus the position of $s_i s_j$ in $G(m,2)$ is $m^2 - s_i^2 + (2(s_i - s_j) - 1) + 1$.

This proves Lemma 1. □

Appendix B

Proof of Lemma 2. For an $(m, 3)$ dBS, the lexicographic list of aperiodic necklaces is given by

Group	m	…	p	…	4	3	2	1
	$m(mm)$	…	$p(pp)$	…	4(44)	3(33)	2(22)	1(11)
	$m, m, m-1$		$p, p, p-1$		443	332	221	
	$m, m, m-2$	…	$p, p, p-2$	…	442	331	211	
	:		:		441	322		
	$m, m, 1$		$p, p, 1$		433	321		
	$m, m-1, m-1$		$p, p-1, p-1$		432	312		
	$m, m-1, m-2$		$p, p-1, p-2$		431	311		
	:		:		423			
	$m, m-1, 1$		$p, p-1, 1$		422			
	:		:		421			
	$m, 1, m-1$		$p, 1, p-1$		413			
	$m, 1, m-2$		$p, 1, p-2$		412			
	:		:		411			
	$m, 1, 1$		$p, 1, 1$					

Let $F(p)$ be the number of elements in group p, $p = 1, 2, 3, \ldots, m$. Thus, we have $F(p) = 1 + 3p(p-1)$.

Since $s_i \geqslant s_j$ and $s_i \geqslant s_k$, there are three possible cases for aperiodic lexicographic necklace s_i, s_j and s_k.

1. If $s_i = s_j = s_k$, the number of elements positioned before $s_i s_j s_k$ is given by

$$\left(\sum_{k=s_i+1}^{m} F(k)\right).$$

2. If $s_i = s_j > s_k$, the number of elements positioned before $s_i s_j s_k$ is given by

$$\left(\sum_{k=s_i+1}^{m} F(k)\right) + 3(s_i - s_k - 1) + 1.$$

3. If $s_i > s_j$ and $s_i > s_k$, the number of elements positioned before $s_i s_j s_k$ is given by

$$\left(\sum_{k=s_i+1}^{m} F(k)\right) + 3(s_i - s_j)(s_i - 1) + 3(s_i - s_k - 1) + 1,$$

   where

$$\left(\sum_{k=s_i+1}^{m} F(k)\right) = \sum_{k=s_i+1}^{m} [1 + 3k(k-1)]$$

$$= \frac{m(m+1)(2m+1) - s_i(s_i+1)(2s_i+1)}{2}$$

$$- \frac{(m - s_i)(3m + 3s_i + 1)}{2}.$$

This proves Lemma 2. □

References

[1] S.R. Yee, P.M. Griffin, Three-dimensional imaging system, Opt. Eng. 33 (1994) 2070–2075.

[2] M.D. Altschuler, B.R. Altschuler, J. Taboada, Topographic mapping of surfaces, Opt. Eng. 20 (1981) 953–961.
[3] P. Vuylsteke, A. Oosterlinck, Range image acquisition with a single binary-encoded light pattern, IEEE Trans. Pattern Anal. Mach. Intell. 12 (1990) 148–164.
[4] J. Posdamer, M.D. Altschuler, Surface measurement by space-encoded projected beam system, Comput. Vision Graphics Image Process. 18 (1981) 1–7.
[5] P.M. Griffin, L.S. Narasimhan, S.R. Yee, Generation of uniquely encoded light patterns for range data acquisition, Pattern Recognition 25 (1992) 609–616.
[6] Y.C. Hsieh, A note on the structured light patterns for three-dimensional imaging systems, Pattern Recognition Lett. 19 (1998) 315–318.
[7] G. Chartrand, O.R. Oellermann, Applied and Algorithmic Graph Theory, McGraw-Hill, Singapore, 1993.
[8] P.L. Emerson, R.D. Tobias, Computer program for quasi-random stimulus sequences with equal transition frequencies, Behav. Res. Meth., Instrum. Comput. 27 (1995) 88–98.
[9] H. Fredricksen, I. Kessler, Lexicographic compositions and de Bruijn sequences, J. Combin. Theory 2 (1981) 63–76.
[10] H. Fredricksen, J. Maiorana, Necklaces of beads in k colors and k-ary de Bruijn sequences, Discrete Math. 23 (1978) 207–210.
[11] A. Ralston, A new memoryless algorithm for de Bruijn sequences, J. Algorithms 2 (1981) 50–62.
[12] H. Fredricksen, A survey of full length nonlinear shift register cycle algorithms, SIAM Rev. 24 (1982) 195–221.

About the Author—YI-CHIH HSIEH was born in Fengyuan, Taichung, Taiwan, on 6 January, 1966. He is a faculty member of Industrial Engineering, National Huwei Institute of Technology. He received a B.B.A. degree in Industrial Management Science from National Cheng Kung University in 1988, an M.S. degree in Applied Mathematics from National Chung Hsing University in 1992, and a Ph.D. degree in Industrial Engineering from the University of Iowa in 1995. His research interests include operations research, mathematical programming, system reliability, and coding theory.