Simple and Robust Binary Self-Location Patterns

Alfred M. Bruckstein, Tuvi Etzion, *Fellow, IEEE*, Raja Giryes, Noam Gordon, Robert J. Holt, and Doron Shuldiner

***Abstract*—A simple method to generate a 2-D binary grid pattern, which allows for absolute and accurate self-location in a finite planar region, is proposed. The pattern encodes position information in a local way so that reading a small number of its black or white pixels at any place provides sufficient data from which the location can be decoded both efficiently and robustly.**

***Index Terms*—de Bruijn sequences, M-sequences, self-location patterns.**

I. Introduction

TAKE a blindfolded man on a random one-hour walk around town and then remove his blindfold. How will he know where he is? He has several options, based on the information he can gather. The man could carefully count his steps and take note of every turn during the blindfolded walk to know his location relative to the beginning of his trip. Armed with a navigation tool such as a sextant or GPS unit, he could ask the stars or the GPS satellites where he is. Finally, he could simply look around for a reference, such as a street sign, a landmark building, or even a city map with a little arrow saying "You are here."

There are numerous applications where a similar problem is encountered. We need to somehow measure the position of a mobile or movable device, using some sort of sensory input. Wheeled vehicles can count the turns of their wheels much like the man counting his steps. Similarly, many devices, from industrial machine stages to ball-mice, employ sensors that are coupled with the mechanics and count small physical steps of a known length, in one or more dimensions. The small relative position differences can be accumulated to achieve *relative self-location* to a known starting point. More recent technologies, such as those found in modern optical mice, use imaging sensors instead of mechanical encoders to estimate the relative motion by constantly inspecting the moving texture or pattern of the platform beneath them.

Sometimes the inherent accumulating error in relative self-location methods, or some other reasons, make them infeasible or unfit for certain applications, where we would want the capability to obtain instant and accurate '*absolute self-location*. Given several visible landmarks of known locations, a mobile robot could calculate its position through a triangulation [1]. Alternatively, cleverly designed space fiducials (e.g., [2]), whose appearance changes with the angle of observation, can also serve for self-location.

Much like street signs for people, there are absolute self-location methods that provide sufficient local information to the device sensors, such that the absolute positioning can be attained. Specifically, planar patterns have been suggested, where a small local sample from anywhere in the pattern provides sufficient information for decoding the absolute position. A naive example could consist of a floor filled with densely packed miniature markings, in which the exact coordinates are literally inscribed inside each marking. Of course, that would require a high sensor resolution and character recognition capabilities. Indeed, there are much more efficient methods, which do with considerably less geometric detail in the pattern. Some commercial products have been utilizing this approach, e.g., a pen with a small imaging device in its tip, writing on paper with a special pattern printed on it, which allows full tracking of the pen position at any time [3].

A classic method for absolute self-location in 1-D is the use of de Bruijn sequences [4], [5]. A de Bruijn sequence of order n over a given alphabet of size q is a cyclic sequence of length q^n, which has the property that each possible sequence of length n of the given alphabet appears in it as a consecutive subsequence exactly once. Thus, sampling n consecutive letters somewhere in the sequence is sufficient for perfect positioning of the sampled subsequence within the sequence. Several methods for the construction of de Bruijn sequences have been proposed, e.g., [5]–[8]. There is also a 2-D generalization, i.e., it is possible to construct a 2-D cyclic array in which each rectangular subarray of a certain size $k \times n$ appears exactly once in the array. These types of arrays are called *perfect maps*, e.g., [9] and [10], and they can serve as the basis for absolute self-location on the plane.

Of special interest and importance in communication are maximal-length linear shift-register sequences known also as M-sequences or pseudonoise sequences [11]. An M-*sequence* of order n is a sequence of length $2^n - 1$ generated by a linear feedback shift-register of length n. In a cyclic sequence of this type, each nonzero n-tuple appears as a window of length n

Manuscript received May 12, 2010; revised December 18, 2011; accepted January 06, 2012. Date of publicationMarch 22, 2012; date of current version June 12, 2012. T. Etzion was supported in part by the United States–Israel Binational Science Foundation, Jerusalem, Israel, under Grant 2006097. R. Giryes was supported by the Azrieli Fellowship. R. J. Holt was supported in part by Bell Laboratories of Alcatel-Lucent.

A. M. Bruckstein, T. Etzion, and R. Giryes are with the Department of Computer Science, Technion, Israel Institute of Technology, Haifa 32000, Israel (e-mail: freddy@cs.technion.ac.il; etzion@cs.technion.ac.il; raja@cs.technion.ac.il).

N. Gordon was with the Department of Computer Science, Technion, Israel Institute of Technology, Haifa 32000, Israel. He is now with the Algorithms Group, Camtek Ltd., Migdal-Haemek 23150, Israel (e-mail: ngordon@cs.technion.ac.il).

R. J. Holt is with the Department of Mathematics and Computer Science, Queensborough, City University of New York, Bayside, NY 11364 USA (e-mail: rholt@qcc.cuny.edu).

D. Shuldiner was with the Department of Mathematics, Technion, Israel Institute of Technology, Haifa 32000, Israel. He is now with the R&D Group, Check Point Software Technologies, Ltd., Tel Aviv 67897, Israel.

Communicated by A. Krzyżak, Associate Editor for Pattern Recognition, Statistical Learning, and Inference.

Digital Object Identifier 10.1109/TIT.2012.2191699

in one period of the sequence exactly once. These sequences have many important and desired properties [11], [12]. A 2-D generalization of such sequences was presented in [12] where they are called pseudorandom arrays. We note also that M-sequences can be used for robust 1-D location by using their error-correction properties, as analyzed in [13].

In this paper, we propose a simple product construction to generate 2-D binary patterns for absolute self-location. This paper is organized as follows. In Section II, we present the product construction based on two sequences with some 1-D window properties. A 2-D array with optimal self-location based on sensing a cross shape is obtained by this construction. In Section III, we prove that the same construction can be used for reasonably effective error-correction of self-location with a rectangular shape. Our conclusions and some interesting problems for future research are presented in Section IV.

II. Proposed 2-D Self-Location Pattern

Our approach for building 2-D arrays with self-location properties is based on a product of two sequences, one of which is a de Bruijn sequence and the other being a sequence in which only half of the patterns appear.

A. Half de Bruijn Sequences

A half de Bruijn sequence of order n is a (cyclic) sequence of length 2^{n-1} which has the property that for each possible n-tuple X, either X or $\bar{X}$ (the bitwise complement of X), but not both, appears in the sequence exactly once as a subsequence.

There are many different ways to construct half de Bruijn sequences. One method, in which a half de Bruijn sequence of length 2^{n-1} is generated from a de Bruijn sequence of order $n-1$ by using the inverse of the well known mapping **D**, called the **D**-morphism, is described in [6]. Another one is based on M-sequences. The following results were proved in [7].

Theorem 1: If $\mathcal{S}$ is an M-sequence of order $n-1$, then for each pair of n-tuples X and $\bar{X}$ either X or $\bar{X}$ appears in $\mathcal{S}$, with the exception of the pair which consists of the all-zero and all-one n-tuples.

Corollary 1: Let $\mathcal{S}$ be an M-sequence of order $n-1$ and let $\mathcal{S}'$ be the sequence obtained from $\mathcal{S}$ by adding another *one* to the unique run with $n-1$ *ones*. Then, $\mathcal{S}'$ is a half de Bruijn sequence.

B. Construction

For two sequences $\mathcal{T} = (t_1, \ldots, t_K)$ and $\mathcal{S} = (s_1, \ldots, s_N)$, the product $\mathcal{T} \otimes \mathcal{S}$ is a $K \times N$ array G in which g_{ij}, $1 \le i \le K$, $1 \le j \le N$, contains the value $t_i \oplus s_j$ (where $\oplus$ denotes modulo 2 addition, also known as the XOR operator).

Take a half de Bruijn sequence $\mathcal{T} = (t_1, \ldots, t_K)$ and a de Bruijn sequence $\mathcal{S} = (s_1, \ldots, s_N)$ of orders k and n, and lengths $K = 2^{k-1}$ and $N = 2^n$, respectively, and let $G = \mathcal{T} \otimes \mathcal{S}$. Clearly, each row in G equals either $\mathcal{S}$ or $\bar{\mathcal{S}}$. Similarly, each column equals either $\mathcal{T}$ or $\bar{\mathcal{T}}$. Thus, each row and each column retain their window property and can serve for self-location in each dimension.

Theorem 2: Each cross-shaped pattern with k vertical and n horizontal entries appears exactly once as a pattern in the array G.

Proof: Let X be a column vector of length k and Y be a row vector of length n. Either X or $\bar{X}$ appears in the sequence $\mathcal{T}$. Let $\mathcal{X}$ be the pattern that appears. Both Y and $\bar{Y}$ appear in the sequence $\mathcal{S}$. Crosses with vertical vector X and horizontal vector Y appear in G only in the portions related to $Z_1 \stackrel{\text{def}}{=} \mathcal{X} \otimes Y$ and $Z_2 \stackrel{\text{def}}{=} \mathcal{X} \otimes \bar{Y}$. Moreover, the crosses in Z_1 are complements of the crosses Z_2. For each cross inside Z_1 and Z_2, there are two possible assignments, depending on the mutual entry of the vertical and horizontal component. Each one of these values appears in either Z_1 or Z_2. ■

By Theorem 2, we can use a cross sensor array to sample k vertical and n horizontal pixels (with one mutual pixel) in order to obtain self-location.

Corollary 2: The proposed method is optimal in terms of the number of sampled pixels required to achieve self-location with a cross of vertical length k and horizontal length n.

Corollary 3: In the array G, each sampled subarray of size $k \times n$ has a unique location.

Remark 1: Similar and more sophisticated product constructions to generate arrays with low redundancy and effective 2-D error-correction capabilities were suggested in various papers, e.g., [14] and [15].

Remark 2: In practice, the planar domain is generally not cyclic. In order to retain the ability to sense all $2^{k-1} \times 2^n$ possible locations with a sensor whose footprint is $k \times n$ pixels array, we extend $\mathcal{T}$ and $\mathcal{S}$ by appending their first $k-1$ and $n-1$ bits, respectively, to their ends. The result is now a $(2^{k-1} + k - 1) \times (2^n + n - 1)$ array.

Example 1: An example of our proposed 2-D grid pattern can be seen in Fig. 1. It was generated using a de Bruijn sequence of order 4 in the horizontal axis, and a half de Bruijn sequence of order 5 in the vertical axis, resulting in a cyclic array of 16×16 pixels. The first column and the first row in the figure contain the location indexes. The second column and the second row contain $\mathcal{T}$ and $\mathcal{S}$, respectively. From the bit values inside the grid, we can decode our position. An example of a sensor readout is marked in the table. The sensor is a 5 by 4 cross. The vertical readout is 10010, and the horizontal readout is 1000 and its unique position can be easily decoded from $\mathcal{T}$ and $\mathcal{S}$.

C. Computing the Location

The first step in our method recovers the 1-D subsequences that correspond to the location in each dimension. Essentially, the 2-D problem is now reduced to two independent 1-D decoding problems. Decoding the location of a subsequence in a de Bruijn sequence is a well-known problem. Decoding of a half de Bruijn sequence is done similarly.

A classic approach of creating a de Bruijn sequence $\mathcal{S}$ of order n requires $\mathbf{O}(n)$ space and $\mathbf{O}(n \cdot 2^n)$ time to generate the whole sequence $\mathcal{S}$ [5], [8]. This involves $\mathbf{O}(n)$ space and $\mathbf{O}(n \cdot 2^n)$

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
		0	0	0	0	1	1	1	1	0	1	1	0	0	1	0	1
1	1	1	1	1	1	0	0	0	0	1	0	0	1	1	0	1	0
2	1	1	1	1	1	0	0	0	0	1	0	0	1	1	0	1	0
3	1	1	1	1	1	0	0	0	0	1	0	0	1	1	0	1	0
4	1	1	1	1	1	0	0	0	0	1	0	0	1	1	0	1	0
5	1	1	1	1	1	0	0	0	0	1	0	0	1	1	0	1	0
6	0	0	0	0	0	1	1	1	1	0	1	1	0	0	1	0	1
7	1	1	1	1	1	0	0	0	0	1	0	0	1	1	0	1	0
8	0	0	0	0	0	①	1	1	1	0	1	1	0	0	1	0	1
9	1	1	1	1	1	⓪	0	0	0	1	0	0	1	1	0	1	0
10	1	1	1	1	①	⓪	⓪	⓪	0	1	0	0	1	1	0	1	0
11	0	0	0	0	0	①	1	1	1	0	1	1	0	0	1	0	1
12	1	1	1	1	1	⓪	0	0	0	1	0	0	1	1	0	1	0
13	1	1	1	1	1	0	0	0	0	1	0	0	1	1	0	1	0
14	1	1	1	1	1	0	0	0	0	1	0	0	1	1	0	1	0
15	0	0	0	0	0	1	1	1	1	0	1	1	0	0	1	0	1
16	0	0	0	0	0	1	1	1	1	0	1	1	0	0	1	0	1

Fig. 1. 16×16 product array of $\mathcal{T} \otimes \mathcal{S}$. The marked cells illustrate a readout by a cross-shaped sensor.

Robust self-location algorithm

The algorithm's input is a rectangle $Z \overset{\text{def}}{=} \{z_{ij} : 1 \le i \le k, 1 \le j \le n\} = (X \otimes Y) \oplus \mathcal{E}$; where X is a vertical k-tuple of a vertical half de Bruijn sequence $\mathcal{T}$; Y is a horizontal n-tuple of a horizontal de Bruijn subsequence $\mathcal{S}$; and $\mathcal{E}$ is a $k \times n$ error pattern. We assume that less than $\frac{n}{4}$ of the bits in each row of $\mathcal{E}$ are *ones* and less than $\frac{k}{2}$ of the bits in each column of $\mathcal{E}$ are *ones*. The output is the original horizontal and vertical subsequences X and Y, respectively.

- Assume that the first bit of X is b. Let D be the first row of Z.
- For each row A of Z
  - if more than half of the bits of $A \oplus D$ are *zeroes* then the corresponding bit of X is b
  - otherwise, the corresponding bit of X is $\bar{b}$.
- Assign 0 or 1 to b to obtain X which appears in $\mathcal{S}$.
- For each column B of Z
  - if more than half of the bits of $B \oplus X$ are *zeroes* then the corresponding bit in Y is a *zero*
  - otherwise, the corresponding bit in Y is a *one*.

Fig. 2. Robust self-location algorithm.

time, with n being the order of the de Bruijn sequence. If running time is an issue, one could create and store in advance a lookup table which lists the locations of all subsequences. This yields $\mathbf{O}(n)$ time complexity, but requires $\mathbf{O}(n \cdot 2^n)$ space for the table. For larger n, a more flexible trade-off between time and space complexity was suggested in [16]. A partial lookup table of evenly spaced locations called *milestones* is created in advance. During runtime, the algorithm which generates the sequence is initialized with the query subsequence and then iterated until one of the milestones is encountered. For example, this can yield $\mathbf{O}(n \cdot 2^{\frac{n}{2}})$ time complexity and will require $\mathbf{O}(n \cdot 2^{\frac{n}{2}})$ space for the table.

In either case, implementation of the self-location process using modern computer systems is feasible, at least for reasonable and practical values of n, depending on the application. Take $n = 16$ for a concrete example. It allows a definition of 2^{16} locations, e.g., a resolution of 0.1 mm over a range of about 6.5 m. In the first approach, it would take, in the worst case, about 65k simple iterations (on a 16-bit register), which can be performed reasonably quickly on current modest embedded processors currently clocked at about tens or hundreds of megahertz. In the second approach, the lookup table would consume about 128 kB (each entry being a two-byte word), which is, again, a quite modest requirement given today's memory capabilities.

There are more efficient methods to generate de Bruijn sequences [17] which can be used in case of an application in which k and n are much larger. The problem of decoding perfect maps was considered, for example, in [18]. A comprehensive survey on this topic was given in [19].

III. Robust Self-Location

The cross-shaped sensor is rather "spread out," so it might be a disadvantage in applications. In this section, we show that this weakness becomes an advantage for robust self-location when the sensor is of a rectangular shape. If we use a $k \times n$ pixel sensor (see Corollary 3), we can utilize the inherent redundancy within the kn bits to decode the location while overcoming a considerable number of faulty bit readings. This is also a very practical choice, considering that 2-D rectangular sensor grids are the most common variety and are the standard choice for most applications.

We assume that less than one quarter of the bits in each row and less than half of the bits in each column of the input array are in error. As will be shown in the sequel, this is a fair assumption which can account for quite strong noise in practical terms. The algorithm for robust self-location presented in Fig. 2 is a simple majority decoding.

Theorem 3: Given a grid of size $2^{k-1} \times 2^n$ and a $k \times n$ pixel sensor, if less than one quarter of the bits in each row and less than half of the bits in each column are in error, then the algorithm accurately decodes the sensor location.

Proof: Since the number of errors in a row is less than $\frac{n}{4}$, it follows that two rows which were originally the same will agree in more than half of their bits. One original row and one complement row will disagree in more than half their bits. Therefore, the related bits of X will be the same or different, respectively. Having all the k bits of X in terms of the variable b, there is only one assignment of a legal k-tuple since the vertical sequence is a half de Bruijn sequence.

Having the correct vertical subsequence X, since the number of errors in a column is less than $\frac{n}{2}$, it follows that if X agrees in more than $\frac{n}{2}$ bits with a column, then the corresponding bit of Y is a *zero*; if it disagrees in more than $\frac{n}{2}$ bits with a column, then the corresponding bit of Y is a *one*. ■

Remark 3: Decoding can be done also if more than one quarter of the bits in some rows are in error. A slightly better condition would be to require that the number of distinct positions in error in any two rows is less than $\frac{n}{2}$. This requirement can be further improved.

A similar algorithm will also work if we will exchange between rows and columns, or equivalently if we will consider a transposed array. Therefore, we can exchange our assumption on the number of wrong bits in a row or a column. But having, for example, at least half of the bits wrong in a given column (or a given row) will cause a wrong identification of the original subsequences.

Lemma 4: Given a grid of size $2^{k-1} \times 2^n$ and a $k \times n$ pixel sensor, if at least half of the bits in one of the columns of a pixel sensor are in error, then we cannot ensure accurate decoding of the original subsequences.

Proof: Let X and Y be two n-tuples which differ only in the first bit. Both X and Y appear as a window of length n in the de Bruijn sequence $\mathcal{S}$ of order n. Let Z be a k-tuple which appears as a window in the sequence $\mathcal{T}$. The products $Z \otimes X$ and $Z \otimes Y$ appear as $k \times n$ windows in the array $\mathcal{T} \otimes \mathcal{S}$. Both $k \times n$ windows differ only in the first column and it would be impossible to distinguish between the two windows if half of the bits in the first column are in error. If more than half of the bits in the first column are in error, then a wrong decoding of the sensor location will be made. The same arguments can be applied to any other column. ■

We note that by Lemma 4, we cannot correct $\lceil \frac{k}{2} \rceil$ or more random errors in a $k \times n$ array. The reason is that the array is highly redundant. This is quite weak from an error-correction point of view. But, by Theorem 3, we are able to correct about $\frac{kn}{4}$ errors in an $k \times n$ array if less than $\frac{n}{4}$ errors occur in a row and less than $\frac{k}{2}$ errors occur in a column. The reason is that redundant rows and columns are used for the majority decoding. This result is quite strong from error-correction point of view. Thus, the weakness for one type of errors becomes an advantage for another type of errors.

Example 2: The 7×9 subarray in Fig. 3 has no more than two errors in a row and three errors in a column. The first row has more than half bits in common with the fifth and the seventh rows. Thus, the vertical pattern is $b\bar{b}\bar{b}\bar{b}b\bar{b}b$. Suppose that $b = 1$, i.e., the vertical pattern is 1000101. We now compare all of the columns with 1000101. If more than half of the corresponding bits agree, the bit in the horizontal sequence is a *one*; otherwise, it is a *zero*. Thus, the sequence is 110111001. The subarray with no errors is presented in Fig. 4.

$$\begin{array}{ccccccccc}
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1
\end{array}$$

Fig. 3. 7×9 subarray with errors.

$$\begin{array}{ccccccccc}
1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array}$$

Fig. 4. Corrected subarray.

TABLE I
PROBABILITIES THAT EACH OF THE ROWS OF AN $n \times n$ WINDOW HAS FEWER THAN ONE QUARTER OF ITS BITS IN ERROR WHEN THE PROBABILITY THAT EACH BIT IS CORRECT IS p

$p \setminus n$	8	16	32	64
0.90	0.191	0.322	0.687	0.9858
0.91	0.253	0.443	0.817	0.9957
0.92	0.329	0.573	0.906	0.9989
0.93	0.417	0.699	0.959	0.9998
0.94	0.517	0.809	0.985	> 0.9999
0.95	0.624	0.894	0.996	> 0.9999
0.96	0.733	0.951	0.9991	> 0.9999
0.97	0.835	0.982	0.9999	> 0.9999
0.98	0.920	0.9962	> 0.9999	> 0.9999
0.99	0.979	0.9997	> 0.9999	> 0.9999

Now, we analyze the error rates in individual bits that allow us to determine the probability that the position is determined correctly. Given a $k \times n$ rectangle in which the probability of each bit being correct is p independent of the other bits, we can determine the probability that each row satisfies the aforementioned condition that less than one quarter of the bits are in error. Then, the probability that each row is satisfactory is the individual row probability raised to the kth power, the number of rows. For simplicity, we assume now that $k = n$.

To find the probability that all of the rows satisfy the row condition, we raise the probabilities $P(n; p)$ to the power of the number of rows. These are given in Table I.

In order to have a probability of at least 0.99 that the row condition is satisfied, we need $p > 0.994$ for $n = 8$, $p > 0.98$ for $n = 16$, $p > 0.95$ for $n = 32$, and $p > 0.91$ for $n = 64$. In order for the row condition to be satisfied with probability at least 0.999, it is sufficient that $p > 0.99$ for $n = 16$, $p > 0.96$ for $n = 32$, and $p > 0.93$ for $n = 64$. Also, if $p > 0.98$ for $n = 32$ or $p > 0.94$ for $n = 64$, the row condition is satisfied with probability greater than 0.9999.

The probability that the column condition (that less than half the bits are in error) is not satisfied when the row condition is satisfied is negligible. For example, if we let $Q(n; p)$ represent the probability that the column condition is not satisfied, i.e.

$$Q(n; p) = \sum_{i=\lceil \frac{n}{2} \rceil}^{n} \binom{n}{i} p^i (1-p)^{n-i} \qquad (1)$$

then we have results such as $Q(16; 0.99) = 1.2 \times 10^{-12}$. In a square array, the probability that the column condition is not satisfied for at least one of the columns is then $1 - (1 - 1.2 \times 10^{-12})^{16} = 1.9 \times 10^{-11}$.

IV. Conclusion and Future Research

Implementing absolute self-location in a planar region using special patterns is a viable and proven approach and can solve a variety of technological problems. In this paper, we have proposed a solution based on robust 2-D arrays with a 2-D window property. The method also has a rather strong error-correction capability. It enables one to correct errors if less than one quarter of the bits in a row and less than half of the bits in a column are in error.

In some applications, the alignment of the sensor array to the grid pattern is not guaranteed. The sensor may be arbitrarily translated and rotated, so that retrieving the local bit matrix is not trivial. Position location of 1-D sequences, when the orientation of the subsequence is not known was considered in [20]. The solution in 2-D arrays is to sample the region at a somewhat higher resolution than k by n, and analyze the image in order to first estimate the pose of the pattern of rows and columns. Since the proposed pattern has a very pronounced structure consisting of identical or inverted rows (as well as columns), this can greatly aid in the task. Using an M-sequence and its complement as vertical and horizontal sequences in our construction can also help in solving of the orientation problem. A complete analysis of these issues is a problem for future research.

There are many other future research problems in this area. Some related to our specific construction and some are to new possible construction methods.

1) As indicated in Remark 3, the claim in Theorem 3 can be strengthened. What is the strongest claim on the error capability of our scheme? Do the de Bruijn sequence and the half de Bruijn sequence that we selected have any influence on this claim?
2) How can we improve the error-correction capabilities of our scheme if the de Bruijn sequence and the half de Bruijn sequence are derived from M-sequences with error-correction capabilities as indicated in [13]?
3) The array obtained by our method can correct a limited number of random errors, even so we proved that the probability for such errors which the method cannot correct is negligible. Generating arrays with window properties which can correct large number of random errors is an important topic for future research.
4) Finally, we note that a folding method for generating pseudorandom arrays from M-sequences was suggested in [12]. This method was subsequently generalized in [21]. Can this method be adapted also to generate better pseudorandom arrays which can correct random errors? Using the M-sequences as suggested by Kumar and Wei [13] for this purpose could be the first step in attempting to find an answer to such questions.

Acknowledgment

The authors would like to thank the anonymous reviewers for their very useful comments which greatly helped us to improve our paper.

References

[1] C. Cohen and F. V. Koss, "A comprehensive study of three object triangulation," in *Proc. SPIE Conf. Mobile Robots*, 1992, pp. 95–106.

[2] A. M. Bruckstein, R. J. Holt, T. S. Huang, and A. N. Netravali, "New devices for 3D pose estimation: Mantis eyes, Agam paintings, sundials, and other space fiducials," *Int. J. Comput. Vis.*, vol. 39, no. 2, pp. 131–139, 2000.

[3] Anoto Group AB Website Switzerland. Lund [Online]. Available: http://www.anoto.com

[4] N. G. de Bruijn, "A combinatorial problem," *Koninklijke Nederlandse Akademie v. Wetenschappen*, vol. 49, pp. 758–764, 1946.

[5] H. Fredricksen, "A survey of full length nonlinear shift register cycle algorithms," *SIAM Rev.*, vol. 24, pp. 195–221, Apr. 1982.

[6] A. Lempel, "On a homomorphism of the de Bruijn graph and its applications to the design of feedback shift registers," *IEEE Trans. Comput.*, vol. 19, no. 12, pp. 1204–1209, Dec. 1970.

[7] B. Arazi, "Method of constructing de Bruijn sequences," *Electron. Lett.*, vol. 12, pp. 858–859, Dec. 1976.

[8] T. Etzion and A. Lempel, "Algorithms for the generation of full-length shift-register sequences," *IEEE Trans. Inf. Theory*, vol. 30, no. 3, pp. 480–484, May 1984.

[9] T. Etzion, "Constructions for perfect maps and pseudorandom arrays," *IEEE Trans. Inf. Theory*, vol. 34, no. 5, pp. 1308–1316, Sep. 1988.

[10] K. G. Paterson, "Perfect maps," *IEEE Trans. Inf. Theory*, vol. 40, no. 3, pp. 743–753, May 1994.

[11] S. W. Golomb, *Shift Register Sequences*. Laguna Hills, CA: Aegean Park, 1982.

[12] F. J. MacWilliams and N. J. A. Sloane, "Pseudo-random sequences and arrays," *Proc. IEEE*, vol. 64, no. 12, pp. 1715–1729, Dec. 1976.

[13] P. V. Kumar and V. K. Wei, "Minimum distance of logarithmic and fractional partial m-sequences," *IEEE Trans. Inf. Theory*, vol. 38, no. 5, pp. 1474–1482, Sep. 1992.

[14] M. Breitbach, M. Bossert, V. Zyablov, and V. Sidorenko, "Array codes correcting a two-dimensional cluster of errors," *IEEE Trans. Inf. Theory*, vol. 44, no. 5, pp. 2025–2031, Sep. 1998.

[15] T. Etzion and E. Yaakobi, "Error-correction of multidimensional bursts," *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 961–976, Mar. 2009.

[16] E. Petriu, "New pseudorandom/natural code conversion method," *Electron. Lett.*, vol. 24, no. 22, pp. 1358–1359, Oct. 1988.

[17] C. J. Mitchell, T. Etzion, and K. G. Paterson, "A method for constructing decodable de Bruijn sequences," *IEEE Trans. Inf. Theory*, vol. 42, no. 5, pp. 1472–1478, Sep. 1996.

[18] C. J. Mitchell and K. G. Paterson, "Decoding perfect maps," *Designs, Codes, Cryptography*, vol. 4, pp. 11–30, 1994.

[19] J. Burns and C. J. Mitchell, M. J. Ganley, Ed., "Coding schemes for two-dimensional position sensing," in *Proc. 3rd IMA Cryptography and Coding Conf.*, Cirencester, U.K., Dec. 1991, pp. 31–66.

[20] Z. D. Dai, K. M. Martin, M. J. B. Robshaw, and P. R. Wild, M. J. Ganley, Ed., "Orientable sequences," in *Proc. 3rd IMA Cryptography and Coding Conf.*, Cirencester, Dec. 1991, pp. 31–66.

[21] T. Etzion, "Sequence folding, lattice tiling, and multidimensional coding," *IEEE Trans. Inf. Theory*, vol. 57, no. 7, pp. 4383–4400, Jul. 2011.

Alfred M. Bruckstein received the B.Sc. and the M.Sc. in Electrical Engineering from the Technion – Israel Institute of Technology, and the Ph.D. in Electrical Engineering from Stanford University, in 1977, 1980 and 1984, respectively. Since 1985, he has been a faculty member at the Technion – Israel Institute of Technology, where he currently a Professor and holds the Ollendorff Technion Chair in Science. He served as the Dean of the Technion Graduate School (2002–2006) and as the Head of Technion's Excellence Program (2007–2012).

Professor Bruckstein held long-term visiting positions at Bell Laboratories, MurrayHill (1987–2000), at TsingHua University, Beijing (2002–2003), and short term visiting positions at several European universities, and is also a Visiting Professor at Nanyang Technological University, Singapore (2009-present). He is on the Editorial Board of the SIAM Journal on Image Analysis, the Journal of Mathematical Imaging and Vision, and the Pattern Recognition Journal, and served in the past on the Editorial Boards of the Imaging Systems and Technology Journal, and the Circuits Systems and Signal Processing Journal. He was involved in the program committees of many conferences and workshops. His research interests are in Image Processing and Analysis, Computer Graphics and Pattern Recognition, and Multi-agent, or Ant Robotics. He is a member of SIAM, AMS and MAA, IAPR.

Tuvi Etzion (M'89–SM'94–F'04) was born in Tel Aviv, Israel, in 1956. He received the B.A., M.Sc., and D.Sc. degrees from the Technion – Israel Institute of Technology, Haifa, Israel, in 1980, 1982, and 1984, respectively.

From 1984 he held a position in the department of Computer Science at the Technion, where he has a Professor position. During the years 1986–1987 he was Visiting Research Professor with the Department of Electrical Engineering – Systems at the University of Southern California, Los Angeles. During the summers of 1990 and 1991 he was visiting Bellcore in Morristown, New Jersey. During the years 1994–1996 he was a Visiting Research Fellow in the Computer Science Department at Royal Holloway College, Egham, England. He also had several visits to the Coordinated Science Laboratory at University of Illinois in Urbana-Champaign during the years 1995–1998, two visits to HP Bristol during the summers of 1996, 2000, a few visits to the department of Electrical Engineering, University of California at San Diego during the years 2000–2011, and several visits to the Mathematics department at Royal Holloway College, Egham, England, during the years 2007–2009.

His research interests include applications of discrete mathematics to problems in computer science and information theory, coding theory, and combinatorial designs.

Dr. Etzion was an Associate Editor for Coding Theory for the IEEE Transactions on Information Theory from 2006 till 2009.

Raja Giryes received the B.Sc degree (Summa Cum Laude) in Computer Engineering and M.Sc degree in Computer Science from the Technion – Israel Institute of Technology, in 2007 and 2009 respectively. Currently, he is a Ph.D. in the Department of Computer Science in the Technion and is an Azrieli fellow. His research interests include image processing, signal processing, sparse approximation theory and compressed sensing.

Noam Gordon received his B.Sc. Cum Laude in Computer Engineering and Ph.D. in Computer Science from the Technion – Israel Institute of Technology, in 2001 and 2010, respectively. Presently he heads the Algorithms Group at Camtek, Israel, researching and developing machine-vision algorithms in automated optical inspection systems for the printed circuit-boards industry. His R&D interests include image processing and analysis, computer vision systems, distributed robotic systems, and swarm intelligence. He further feeds his soul with playing the piano and practicing artistic photography.

Robert J. Holt has been a member of the faculty of the Department of Mathematics and Computer Science at Queensborough, City University of New York, since 2003. Professor Holt received the B.S. in Mathematics from Stanford University in 1982 and the Ph.D. in Mathematics from the Massachusetts Institute of Technology in 1986. He taught in the Mathematics Department at Northwestern University from 1986 to 1988, and was a researcher in Bell Laboratories of AT&T and Lucent Technologies, now Alcatel-Lucent, from 1988 to 2003. He is the author of over forty papers, mainly in the field of computer vision.

Doron Shuldiner received the B.Sc. degree in mathematics from the Technion – Israel Institute of Technology, Haifa, Israel, in 2008. He is working now in the industry on network engineering.