*Sensors and Materials*, Vol. 29, No. 6 (2017) 645–657
MYU Tokyo

S & M 1357

# Finite Geometry Permutation Decoding for Wireless Sensor Network Applications

Jyun-Jie Wang and Chi-Yuan Lin*

Department of Computer Science and Information Engineering, National Chin-Yi University of Technology,
Taichung 41170, Taiwan, ROC

(Received September 27, 2016; accepted December 5, 2016)

***Keywords:*** Euclidean geometry code, energy efficiency, sensor network, network lifetime, majority logic decoding, permutation

In this study, we propose a novel coding method, called geometry permutation decoding (GPD), for square sensor topologies. The GPD algorithm provides a two-dimensional geometry construction and decodes information sequences based on this construction. The two-dimensional construction for square wireless sensor networks (WSNs) is suitable for employing channel coding, especially product codes and finite geometry codes, and for detecting multiple error patterns. A majority logic decoding (MLD) is a less correcting efficient but simpler-to-implement decoding method. To improve the drawback of the less correcting capacity, the GPD algorithm not only gives better bit error rates at low decoding complexity but also provides a new solution suitable for decoding in two-dimensional WSNs. In particular, this decoding work in the two-dimensional WSNs is achieved over the bit level without a packet level so that the overhead in the network layer can be reduced.

## 1. Introduction

In his landmark paper in 1948, Shannon proved the existence of the limited bound over a noisy channel.(1) Since Shannon's work, much effort has been focused on the problem of devising efficient and fast encoding and decoding algorithms for error control in a noisy environment. Recent developments have contributed toward achieving reliability while diminishing complexity. In 1963, Massey(2) proposed a less efficient but simpler-to-implement decoding method called threshold decoding. This method spawned some practical applications of linear codes to digital transmission over telephone, satellite, and radio channels. Shortly thereafter, researchers also proposed a number of decoding algorithms that were relatively easy to implement for hard-decision decoding of cyclic codes. Majority logic decoding (MLD)(3) is a simple and effective method for decoding code classes in particular, such as cyclic codes. Most codes with MLD found so far are cyclic codes. One of the major cyclic codes is a code constructed based on finite geometries, called Euclidean geometries. These codes are well-known finite geometry codes. Finite geometry codes were first investigated by Rudolph in 1967.(3) Another special subclass of finite geometries codes, called Euclidean geometry (EG) low-density parity-check codes, was introduced by Kou and colleagues(4,5) in 2000. MLD algorithms are hard-decision decoding algorithms that can be easily implemented. MLD based on orthogonal parity check sums from the parity check matrix of a code

*Corresponding author: e-mail: chiyuan@ncut.edu.tw

http://dx.doi.org/10.18494/SAM.2017.1477

ISSN 0914-4935 © MYU K.K.

is a simple decoding method.[6] The error-correcting capability of majority logic decodable codes is based on the number of check sums. These codes are nearly capable of correcting the error bits, which are about half the number of the check sums. The parity check sums orthogonal on one bit can be used to estimate that bit or to decode that received digit. The simplest and fastest decoding algorithm for error-correcting codes is the one-step MLD algorithm,[7,8] but it provides the worse decoding efficiency compared with other decoding algorithms for error-correcting codes.

In this study, we also propose a novel decoding method based on parity check sums, but the method is not majority-logic-based. This method uses error patterns of 2-flat or a plane as an alternative to orthogonal check sums. There are many lines through one point with various gradients, i.e., independent lines, in a plane. A plane can be constructed by two of these independent lines. These planes constructed by independent lines lead to permutations of the received sequence and provide a variety of error patterns in two dimensions. This work locates some error bits according to these error patterns as long as error bits are less than three in number. Proposed in this article is a method called the geometry permutation decoding (GPD). In this study, the GPD algorithm is suitable for application to wireless sensor networks (WSNs) with clustering protocols. Network lifetime is an important issue in WSNs. To achieve efficient network lifetime, one of the efficient protocols is the clustering scheme. Clustering schemes have some cluster heads (CHs) and some node numbers within a cluster. CHs are fusion points for data aggregation, so that the actual data sent to the sink is reduced. The CHs closest to the sink are burdened with a heavy relay traffic load, and they run out of power first.[9] A merging of clustering and multi-hop communication can effectively overcome the path loss effects.[10,11] In general protocols, the packets are transmitted in network access, and the decoding scheme is often achieved in a medium access control layer. Instead, an efficient decoding method using finite geometry constructed over the bit level is proposed, called the GPD algorithm. This scheme uses a localization technique in two dimensions constructed to correct the error patterns without employing packets transmitted in the network.

The rest of this paper is organized as follows. In Sect. 2, we state our problem in wireless sensor networks. In Sect. 3, we briefly describe our major work on the proposed GPD. In Sect. 4, we provide experimental results and constructive discussions. Finally, we state our conclusions in Sect. 5.

## 2. Problem Statement

In this section, we present a statement of the proposed problem and its solution. Assume that there are three events in a two-tier data dissemination protocol shown in Fig. 1.

In Fig. 1, the field is partitioned into a grid of cells. These sensor nodes are classified into sixteen groups, and the CHs are located in the intersections on the grid. Each cell is a $d \times d$ square area. One CH propagates data announcements to the others. Consider three events represented by a square on the grid, and the position of the CH for the three events is given by

$$\{(x + id, y + jd) \,|\, i, j \in Z^{+}\}, \tag{1}$$

where $(x,y)$ represents the coordinates for the events. In Fig. 2, the sink starts by flooding its query with its primary close to a CH node. This CH records the location through these nodes and forwards the query to its neighbour CH until the query reaches the events. The packets are

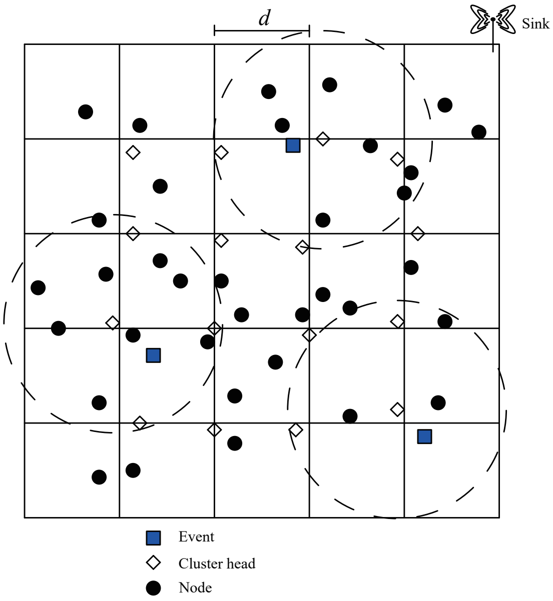

Fig. 1. (Color online) An example of a sensor network.

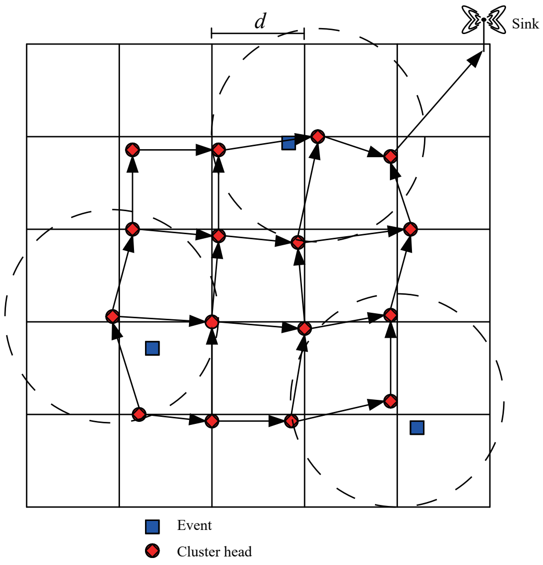

Fig. 2. (Color online) Two-dimensional clustering topology for sensor network.

returned to the CH close to the sink along the path that the query traverses. Finally, the CH close to the sink forwards the packets to the sink through sensor nodes. Low-energy adaptive clustering hierarchy (LEACH)(4) is one of the first clustering schemes, which plays a key role in reducing energy consumption and enhancing the network lifetime. LEACH can provide a balance of energy consumption between these nodes in the field. In Fig. 2, the LEACH protocol can be achieved for network access.

Assume that the sink in Fig. 2 is located in the centric field shown in Fig. 3 and the two-dimensional tree topology in Fig. 3 is used. In WSN, the cooperation between sensor nodes is an important ideal. Instead of working in packet level, this study proposes a decoding method achieved in bit level. This decoding scheme requires the cooperation between the CHs in Fig. 3, and consumes energy $l'E_t$, where $E_t$ and $l'$ denote the energy of one bit and the number of CHs, respectively. Assume that a packet consists of $l$ bits and the energy consumption is $l'E_t$ for a packet transmission. Generally, $l$ is considerably larger than $l'$. Unlike the case with the general LEACH in this study, the transmitting data is not forwarded to the sink but uses the tree-routing topology as the routing scheme. In addition, each sensor node sends exactly one bit to the sink without transmitting packets. The aim of these modifications is to improve the power consumption.

The proposed WSN model shown in Fig. 3 is a two-dimensional grid configuration, and the node close to the intersection in the two-dimensional WSN is to be a CH. LEACH done in Refs. 9 and 12 is a common cluster-based communication model for WSNs. Because the proposed model is also a cluster-based WSN, LEACH is suitable for application to this work and we use a similar radio model as noted in Refs. 9 and 12. In this case, the distance between the transmitter and the receiver for sending one bit is $d$, and the power consumption is given by

$$E_t = \begin{cases} E_{elec} + \varepsilon_{fs} d^2, \text{if } d \le d_0 \\ E_{elec} + \varepsilon_{tr} d^4, \text{if } d \ge d_0 \end{cases}, \tag{2}$$

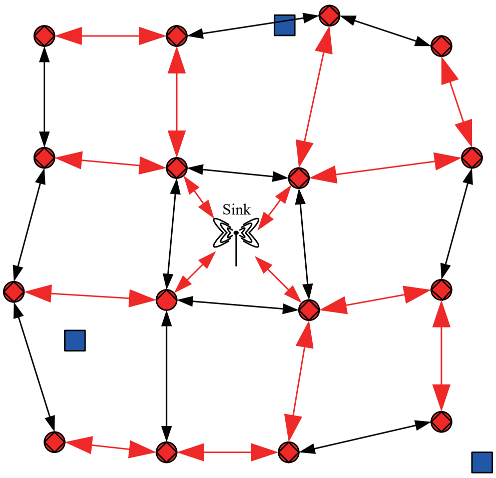

Fig. 3. (Color online) The sink is located at the centre of a two-dimensional network.

where $E_{elec}$ is the energy consumption of the wireless sensor node for transmitting and receiving bit data, and $d_0 = \sqrt{\varepsilon_{fs}/\varepsilon_{tr}}$. Both the parameters $\varepsilon_{fs}$, free space, and the $\varepsilon_{tr}$, multiple path fading channel models, vary according to the distance between a sender and a receiver. The cost of energy for the receiver is given by

$$E_r = E_{elec}. \tag{3}$$

We assume an area $m \times m$ m$^2$, and $N$ is the number of nodes uniformly distributed over the square area. The sink is at the centre of the two-dimensional network. Each node in the cluster transmits one bit of data to the elected CH node. The energy dissipated in the CH node during a round is given by(13)

$$E_{clu} = \frac{N}{k} \times E_{elec} + \frac{N}{k} \times E_d + \varepsilon_{fs} \times d_{sink}^2, \tag{4}$$

where $k$ denotes the number of clusters, $E_d$ is the processing cost of a bit report to the sink, and $d_{sink}$ is the average distance between a CH and a sink. The other sensor nodes are given by

$$E_{sen} = E_{elec} + \varepsilon_{fs} \times d_{ch}^2, \tag{5}$$

where $d_{ch}$ is the average distance between a cluster member and its CH, which is given by

$$d_{ch}^2 = \frac{M^2}{2\pi k}. \tag{6}$$

According to this network model, the difference in the power consumption can be evaluated between the packet and the bit level. When the lengths of the packet are quite large, the protocols for packet access consume more power. Then, each sensor node generates a random number with a value between 0 and 1 and compares it to a predefined threshold $T(n)$. If the random value is less than $T(n)$, the sensor node becomes a CH in that round; otherwise, it is a cluster member, and

$$T\left(n\right) = \frac{P}{1 - P\left(r \bmod \frac{1}{P}\right)} \forall n \in G, \tag{7}$$

where $P$ is the percentage of CHs over all nodes in the network, $r$ is the number of rounds of selection, and $G$ is the set of nodes that are not selected in round $1/P$.

In the next section, a new decoding scheme using finite geometry construction over bit level is proposed. Our work results in low power consumption, along with the method discussed in Sect. 2, because the consumption of the proposed method is inversely proportional to the length of the packet.

## 3. Decoding Using Two-Dimensional Error Patterns

In this section, we discuss the proposed method and a number of fundamental finite geometries.

### 3.1 Permutation of error patterns

Consider all the roots $\{\alpha_i\}$ in the finite field $F_{2^s}^m$. The collection of the following $2^s$ points, $\left\{\alpha_i + \alpha_j\beta \mid \beta \in F_{2^s}\right\}$, with $\beta \in F_{2^s}$ constitutes a line in $EG(m,2^s)$, which passes through the point $\alpha_i$. There are $(2^{ms} - 1)/(2^s - 1)$ lines in $EG(m,2^s)$, which intersect at $\alpha_i$. Moreover, $\{\alpha_j\beta\}$ and $\{\alpha_i + \alpha_j\beta\}$ do not have any common points, and these are called parallel lines. In $EG(m,2^s)$, for every line passing the origin, there are $2^{ms-s}$ parallel lines shown in Fig. 4.

Let $L_1 = \{\alpha^i + \alpha_1\beta_1\}$ and $L_2 = \{\alpha^i + \alpha_2\beta_2\}$ be the two lines that have point $\alpha^i$ in common. A new plane consisting of the two lines is defined by

$$P_{1,2} = L_1 \oplus L_2 = \left\{\alpha^i + \alpha_1\beta_1 + \alpha_2\beta_2\right\}. \tag{8}$$

Because there are $(2^{ms} - 1)/(2^s - 1)$ lines through one point, the number $C_2^{(2^{ms}-1)/(2^s-1)}$ of planes can be constructed. Each plane consists of two parallel bundles shown in Fig. 5.

The plane can be checked by eight lines and the obtained eight parity check equations. The GPD algorithm uses these check sums to locate the error patterns. Therefore, these planes constructed by lines through one point are permuted to a variety of two-dimensional matrices and the error patterns correspond to these planes. To carry out the GPD method, this work employs EG codes of length $2^{ms} - 1$. We construct an EG code with a parity check matrix whose null space contains the incidence vectors of all the lines of $EG(m,2^s)$, which do not pass through the origin. Let $\alpha$ be a primitive element of the finite field $F_{2^s}^m$. For example, let $m = 3$, $s = 2$, and $\mu = 0$. The finite field $F_{2^2}^3$ may be regarded as the $EG(3,2^2)$ over $F_{2^2}$. Let $\alpha$ be a primitive element in $F_{2^2}^3$. Let $h$ be a nonnegative integer less than 63. The generator polynomial $g(x)$ of the (0,2)th-order EG code of length 63 has $\alpha^h$ as a root if and only if $0 < \max_{0 \le l < 2} W_{2^2}\left(h^{(l)}\right) \le 3$ Hence, the (0,2)th-order EG code of length 63 is a (63,13) cyclic code. In this work, we show that this code has total $C_2^5 = 10$ planes

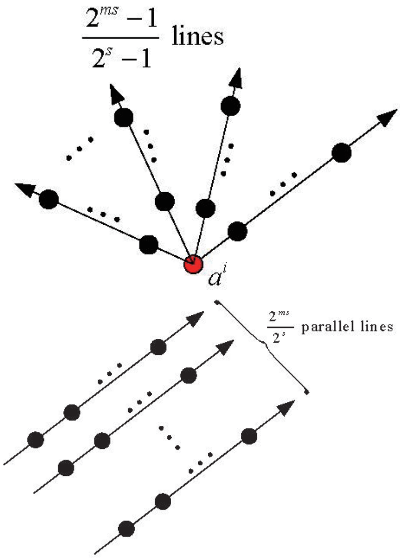
$\frac{2^{ms}-1}{2^s-1}$ lines
$a^i$
$\frac{2^{ms}}{2^s}$ parallel lines

Fig. 4. (Color online) The point intersects on one point and parallel lines.

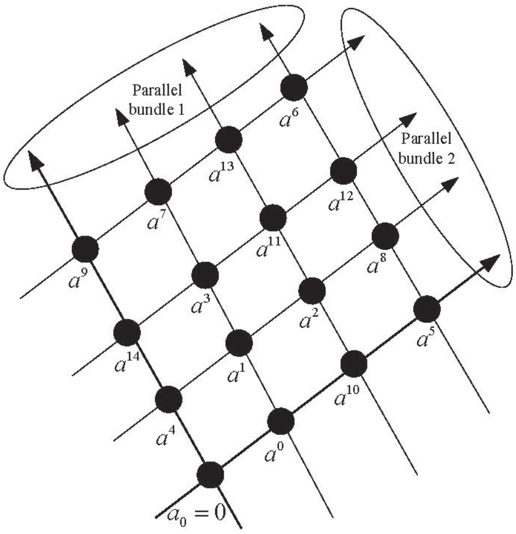

Fig. 5. The plane $P$ consists of two parallel lines.

consisting of a variety of two independent bases in some plane. Consider the plane $P \in EG\left(3, 2^2\right)$ that passes through the point $\alpha^{44}$:

$$P = \begin{bmatrix} \alpha^{44} & \alpha^{4} & \alpha^{35} & \alpha^{39} \\ \alpha^{8} & \alpha^{61} & \alpha^{53} & \alpha^{54} \\ \alpha^{24} & \alpha^{37} & \alpha^{43} & \alpha^{13} \\ \alpha^{10} & \alpha^{48} & \alpha^{36} & \alpha^{51} \end{bmatrix}. \tag{9}$$

There are $(2^4 - 1)/(2^2 - 1) = 5$ lines in the plane $P$, and these lines are through point $\alpha^{44}$. Two independent lines can be constructed as a two-dimensional plane, and the number of these planes is 10 as follows:

$$\begin{aligned} &\begin{cases} L_1 = \left(\alpha^{44} & \alpha^{4} & \alpha^{35} & \alpha^{39}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{8} & \alpha^{24} & \alpha^{10}\right) \end{cases}, \begin{cases} L_1 = \left(\alpha^{44} & \alpha^{53} & \alpha^{13} & \alpha^{48}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{39} & \alpha^{4} & \alpha^{35}\right) \end{cases}, \begin{cases} L_1 = \left(\alpha^{44} & \alpha^{37} & \alpha^{36} & \alpha^{54}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{39} & \alpha^{4} & \alpha^{35}\right) \end{cases} \\ &\begin{cases} L_1 = \left(\alpha^{44} & \alpha^{61} & \alpha^{43} & \alpha^{51}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{39} & \alpha^{4} & \alpha^{35}\right) \end{cases}, \begin{cases} L_1 = \left(\alpha^{44} & \alpha^{53} & \alpha^{13} & \alpha^{48}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{10} & \alpha^{8} & \alpha^{24}\right) \end{cases}, \begin{cases} L_1 = \left(\alpha^{44} & \alpha^{37} & \alpha^{36} & \alpha^{54}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{10} & \alpha^{8} & \alpha^{24}\right) \end{cases} \\ &\begin{cases} L_1 = \left(\alpha^{44} & \alpha^{37} & \alpha^{36} & \alpha^{54}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{10} & \alpha^{8} & \alpha^{24}\right) \end{cases}, \begin{cases} L_1 = \left(\alpha^{44} & \alpha^{37} & \alpha^{36} & \alpha^{54}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{53} & \alpha^{13} & \alpha^{48}\right) \end{cases}, \begin{cases} L_1 = \left(\alpha^{44} & \alpha^{61} & \alpha^{43} & \alpha^{51}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{53} & \alpha^{13} & \alpha^{48}\right) \end{cases}, \\ &\text{and} \begin{cases} L_1 = \left(\alpha^{44} & \alpha^{61} & \alpha^{43} & \alpha^{51}\right) \\ L_2 = \left(\alpha^{44} & \alpha^{37} & \alpha^{36} & \alpha^{54}\right) \end{cases}. \end{aligned} \tag{10}$$

Assume that there are three error bits in the plane $P$. All error patterns permuted by these independent lines are illustrated in Fig. 6. In Fig. 6, each error pattern has a row and a column as a syndrome pair. These syndrome pairs can be divided into (1, 3), (3, 1), (3, 3), and (1, 1). In the important case when the syndrome pair is (3, 3), the three error bits can be identified according to a row and a column syndrome. In Fig. 6, the second pattern of the top corresponds to $\alpha^{44}$, $\alpha^{24}$, and $\alpha^{51}$. Figures 7 and 8 are the error patterns for two error bits and one error bit, respectively.

Table 1 shows the all error patterns for which the number of error bits is less than 4. If the number of error bits is equal to 0 or 1, the total number of error patterns is 10. However, the situation in which the number of error bits is more than 1 is characterized by a variety of syndrome numbers, including (4, 6), (3, 6, 1), and (1, 4, 6).

The decoding procedures for the GPD algorithm based on the error patterns tabulated in Table 1 can be achieved using the tree construction shown in Fig. 9. The decoding procedures are divided into three steps. The first is to estimate the category of the syndrome number. The category of the syndrome number may be even or odd. If the syndrome of an even number is found, the decoding

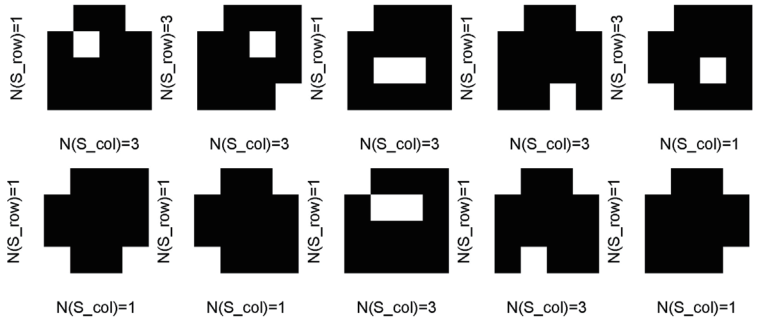

Fig. 6. Error patterns for three error bits in plane *P*.

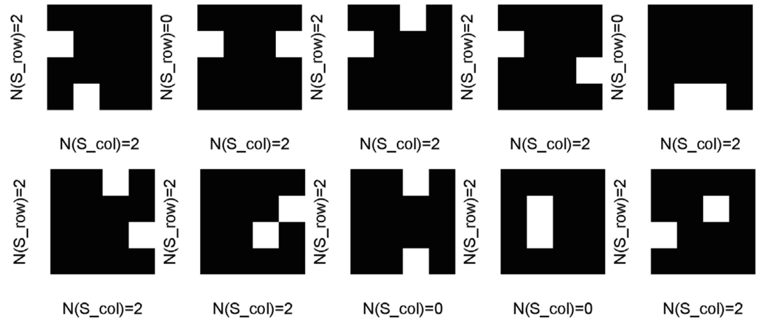

Fig. 7. Error patterns for two error bits in plane *P*.

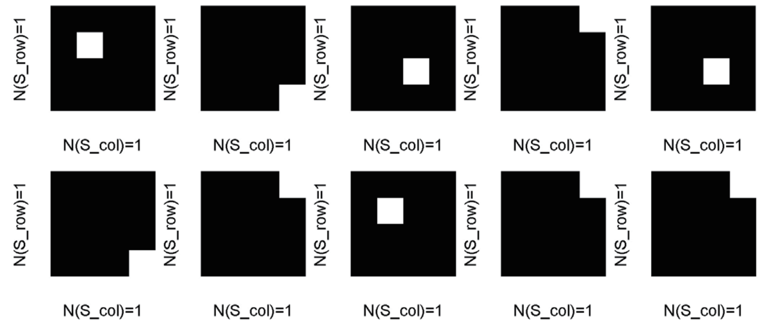

Fig. 8. Error patterns for one error bit in plane *P*.

Table 1
Error pattern for error bits less than 4.

| Number of error bits | Error pattern | | Number of total syndrome | |
|---|---|---|---|---|
| | Row syndrome | Column syndrome | | |
| 0 | 0 | 0 | 10 | 10 |
| 1 | 1 | 1 | 10 | 10 |
| 2 | 2 | 0 | 4 | 10 |
| | 2 | 2 | 6 | |
| 3 | 1 | 1 | 3 | 10 |
| | 1 | 3 | 6 | |
| | 3 | 3 | 1 | |
| 3 | 1 | 3 | 4 | 10 |
| | 3 | 3 | 6 | |

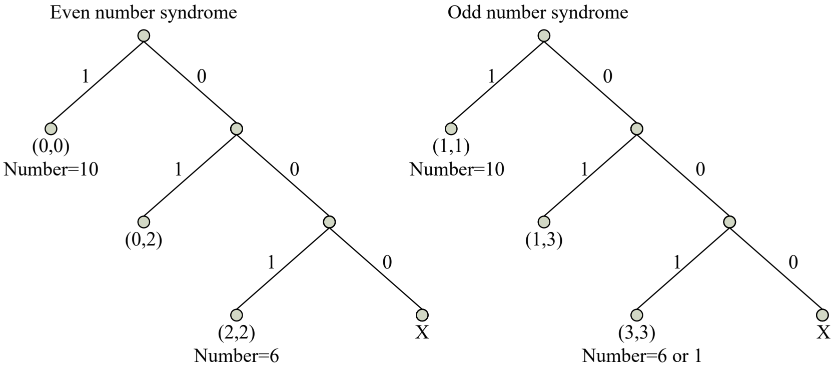

Fig. 9. Decoding tree for the GPD algorithm.

tree is the one drawn on the left in Fig. 9. The decoding tree constructed by odd syndrome patterns is similar to that of even patterns. Although the GPD algorithm finds only the syndrome patterns for which the number of error bits is less than 4, the decoding plane can be randomly chosen in the $EG(m,2^s)$. Because of the random property of error bits, the decoding plane chosen randomly can be regarded as a pseudo-random interleaver. If the decoding algorithm is overall in $EG(m,2^s)$, a complete decoding procedure would be prohibitively complex; however, because more than one decoding plane is used, it is possible to employ a simple decoding procedure for each 2-flat space in an iterative fashion, in which the soft decision is passed to the other.

The implementation of this decoding scheme can be simply achieved using memory and registers. The decoding realization is implemented in two steps. A message bit of a CH is first arranged as shown in the memory of Fig. 10. It is possible to construct a memory by interleaving, that is, simply by arranging data from CHs into the rows of the memory and then permuting the data in the memory as another decoding array. At the first step of encoding, each row of the memory array is encoded into a parity check sum. At the second step of encoding, each of the columns of the memory formed in the first encoding step is encoded into a parity check sum for all the columns. For the example in Fig. 6, a pattern of three errors at the distributed corners of a triangle gives two errors in each of the two rows or columns, and the other may be in any one row or column. This error pattern is not correctable by simple correction on rows and columns.

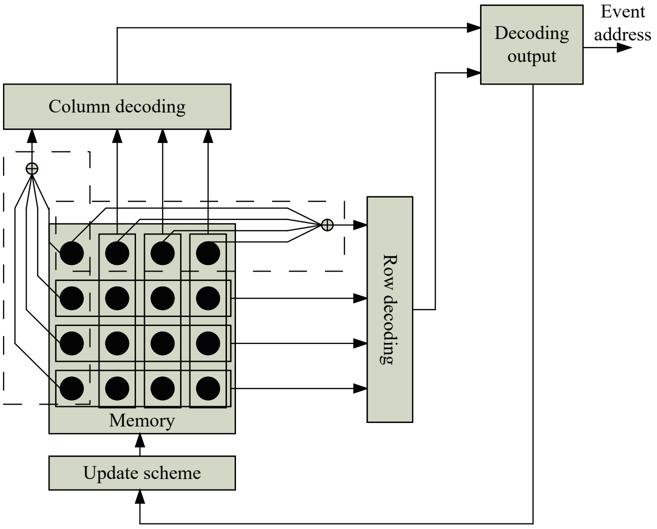

Fig. 10. Implementation of two-dimensional decoding.

Nevertheless, a simple permutation in a variety of two-dimensional memory composed of two lines in finite geometry can be very effective.

Permuting memory constructed by finite geometry can easily be generalized by interleaving several different codes of the same length. The decoding efficiency depends on the decoding capacity required by different codes.

### 3.2 Computational complexity

Although the parity check matrix for EG codes forms a sparse matrix, the decoding algorithm can use a sum-product algorithm to achieve good decoding error probabilities. EG codes with SPA algorithms suffer a disadvantage in decoding complexity in high-dimensional space. This study proposed a way of improving the issue of decoding algorithms using subconstruction over $EG(m, 2^s)$. If the $EG(m, 2^s)$ is large, the dimension of the plane is still sufficiently small to enable low-complexity decoding. The parity check sums can also be found using the constructions of the lines in the plane.

The method proposed in this study can reduce the decoding complexity more than other decoding algorithms. The number of parity check sums for MLD decides the complexity associated with the line construction in $EG(m, 2^s)$. The decoding in some points for MLD needs $(2^{ms} - 1)/(2^s - 1)$ lines, and therefore MLD can be capable of correcting $[(2^{ms} - 1)/(2^s - 1)/2]$ bits. The number of check sums is also equal to $[(2^{ms} - 1)/(2^s - 1)/2]$. The complexity of GPD is required to locate the check sums of two dimensions. However, the GPD algorithm requires providing $C_2^{(2^{ms}-1)/(2^s-1)}$ check sums of two dimensions to locate the error patterns. The number for finding the check sums for the MLD algorithm is

$$C_{MLD}(m, s) = (2^{ms} - 1)^2/(2^s - 1). \tag{11}$$

For the GPD algorithm, the complexity is

$$C_{GPD}(m, s) = 2^{s+1} C_2^{(2^{ms}-1)/(2^s-1)}. \tag{12}$$

Moreover, the proposed method can also be decoded using iteration decoding procedures. After the partial received sequence is decoded using the GPD algorithm, we can choose the decoding plane again to decode based on the GPD algorithm. A GPD with an iterative procedure not only further improves the decoding error probability but also controls the complexity of the GPD algorithm.

## 4. Simulation Results

The experimental results demonstrate how the decoding performance for the GPD algorithm in a wireless sensor network works. Consider a wireless sensor network consisting of 16 clusters uniformly dispersed over a field to continuously monitor the environment, as drawn in Sect. 2. The sink is located in the center of the field. These sensor nodes are location-unaware and have the same capabilities and resources. Each node in the field also has a unique identifier. Nodes can act in inactive node or sleep mode. First, when three events occur in the field at the same time, the sensor node closest to these events transmits bit data to the CH node. All CHs broadcast their data to other CHs. Then, each CH has the data of the remaining CHs. The CHs can evaluate the collected data using the GPD algorithm and decode them. Thus, the processor in the CH is capable of decoding these three error patterns resulting in the channel. This study uses a simplified mode, as noted in Ref. 9, for communication energy consumption. This study works at the bit level instead of packet level. A network of 100 sensor nodes deployed uniformly over an area of size (100 × 100 m$^2$) with a sink located at position (50, 50) is designed. For simplification, an ideal MAC layer and binary symmetric channel are considered. The environmental parameters are listed in Table 2.

In the simulation, we assume a wireless sensor network consisting of number nodes and cluster nodes. The cluster members use a single point to communicate with the CH node. Figure 11 shows that the number of dead nodes per round increases as the number of rounds increases.

Table 2
Propagation characteristics of simulation parameters.

| Parameter | Value | Unit |
|---|---|---|
| Location of sink | (50,50) | m |
| $N$ | 100 | number |
| $E_{elec}$ | 0.05 | µJ/bit |
| $d_0$ | 86 | m |
| $\varepsilon_{fs}$ | 10 | pJ/bit/m$^3$ |
| $\varepsilon_{tr}$ | 1.3 | nJ/bit/m$^4$ |
| $E_r$ | 0.05 | µJ/bit |
| $E_{ori}$ | 500 | mJ |

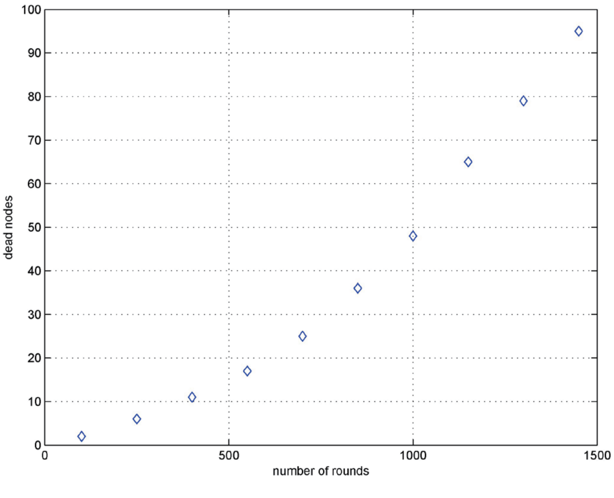

Fig. 11. Number of dead nodes per round.

Here, we only simulate the energy consumption of the proposed decoding method for our sensor network, and the other environment parameters are the same as those of the general set. This work can achieve the network lifetime until close to 1500 rounds when performing the simulation for the GPD algorithm. Figure 12 shows that the energy consumption of one CH node runs down until it is dead. The energy consumption of a cluster node may be different depending on the topology.

In the second experiment, the first task was to address the complexity problem of constructing finite geometry codes with GPD decoding, as shown in Table 3. In the experiment, the GPD algorithm was compared with the MLD algorithm. The operator of check sums for the GPD and MLD algorithms is given by $2^{2s} \times C_2^{(2^{ms}-1)/(2^s-1)}$ and $(2^{ms} - 1)^2/(2^s - 1)$, respectively. The simulation results are shown in Table 3.

The functions of *m* and *s* determine the code size. When the size of finite geometry is $m > 4$ and $s > 3$, the experimental results are difficult to simulate and implement owing to their high complexity. Table 3 shows that the check sums of the GPD algorithm are similar to those of the MLD algorithm, i.e., both complexities are comparable. The second task was to demonstrate simulation results using the finite geometry code. Let $m = 2$, $s = 2$, and $\mu = 0$. The finite field $F_{2^2}^2$ may be regarded as the $EG(2, 2^2)$ over $F_{2^2}$, and the generator polynomial of the finite geometry code of length 15 is $g(x) = 1 + x^4 + x^6 + x^7 + x^8$. This code is an (15, 7) Bose Ray-Chaudhuri Hocquenghem (BCH) code and the two-dimensional EG code is the simplest finite geometry code. This code has a minimum distance of 5 and rate $R = 0.4667$. The error performance with the MLD and GPD algorithms is shown in Fig. 13. We see that MLD and GPD provide for less coding gain over the binary channel but require the least decoding complexity. With GPD, this code is capable of correcting 3 or fewer errors. The MLD gives less error performance than the GPD. In fact, the GPD can use iteration decoding to enhance the error performance. This method offers a good trade-off between error performance and decoding complexity. From Fig. 13, we see that the GPD achieves about 0.2 dB coding gain over the MLD at various bit error rates (BERs) with some additional computational complexity.

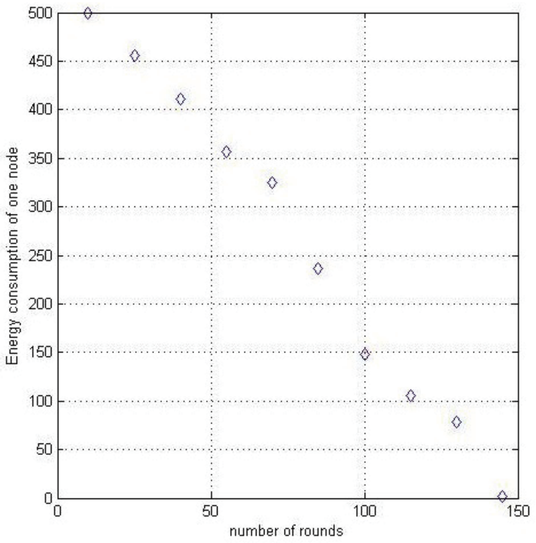

Fig. 12. Energy consumption of one node as a function of number of rounds.

Table 3
Comparison of computational complexity between MLD and GPD.

| *m* | *s* | GPD | MLD |
|---|---|---|---|
| 2 | 2 | 80 | 75 |
| 2 | 3 | 576 | 567 |
| 2 | 4 | 4352 | 4335 |
| 2 | 5 | 33792 | 33759 |
| 4 | 2 | 1680 | 1323 |
| 4 | 3 | 42048 | 37303 |
| 4 | 4 | 1188096 | 1117935 |
| 5 | 2 | 28560 | 21675 |
| 5 | 3 | 2733120 | 2395575 |

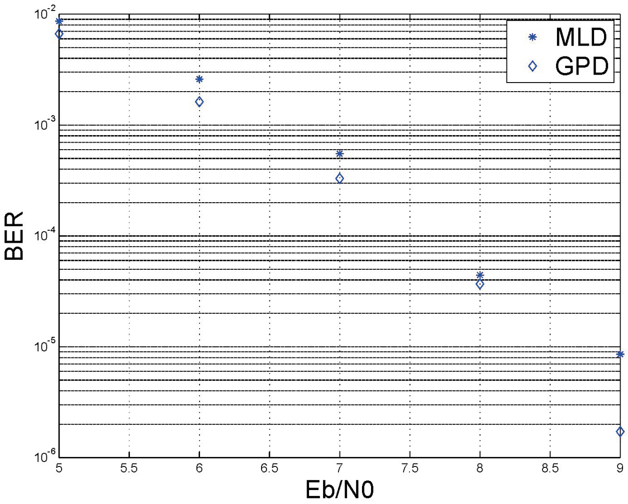

Fig. 13. (Color online) BER for finite geometry over $EG(2,2^2)$ between MLD and GPD.

## 5. Conclusions

In this report, we propose a novel decoding method based on partial geometry construction for error control coding in a grid-based sensor network. Generally, the WSNs transmit data using a packet form. In this study, the proposed method was used in bit level, and the simulation of energy consumption was run using bit streams. We used 2-flat construction and two-dimensional check sums to locate error patterns. The proposed method has not only low decoding complexity but also good error efficiency for hard decoding. An iterative decoding procedure was used in this work. Moreover, the proposed methods reduced the operational complexity for error control coding and provided superior BER performance in comparison with other decoding algorithms.

## Acknowledgment

This research was supported by the National Chin-Yi University of Technology (NCUT), Taichung, R.O.C., under Contract No. 16-R-CC-015.

## References

1 C. E. Shannon: Bell Syst. Tech. J. **27** (1948) 379.
2 J. L. Massey: Threshold Decoding (MIT Press, Cambridge, 1963).
3 L. D. Rudolph: IEEE Trans. Inf. Theory **IT-13** (1967) 305.
4 Y. Kou, S. Lin, and M. Fossorier: Proc. IEEE Intl. Symp. Inf. Theory (Sorrento, June 25–30, 2000) pp. 2711–2736.
5 R. Lucas, M. Fossorier, Y. Kou, and S. Liu: IEEE Trans. Commun. **48** (2000) 931.
6 K. J. C. Smith: Institute of Statistics Mimeo Series, No. 561 (University of North Caapel Hill, 1967).
7 S. Lin and G. Markowsky: IBM J. Res. Dev. **24** (1980) 56.
8 L. Q. Zeng, L. Lan, Y. Y. Tai, B. Zhou, S. Lin, and K. Abdel-Ghaffard: IEEE Trans. Commun. **56** (2008) 378.
9 W. Heinzelman, A. Chandrakasan, and H. Balakrishnan: IEEE Trans. Wireless Commun. **1** (2002) 660.

10 N. Israr and I. Awan: Int. J. Distrib. Sens. Netw. **4** (2008) 176.
11 K. Nawaz and A. P. Buchmann: IJWMN **2** (2010) 18.
12 D. Kumar: IET Wireless Sens. Syst. 4 (2014) 9.
13 D. Kumar, T. Aseri, and R. Patel: Comput. Commun. **32** (2009) 662.

## About the Authors

**Jyun-Jie Wang** received his B.S. degree in Electronic Engineering from National Chi-Yi University of Science and Technology, Taiwan, in 2003, and his M.S. and Ph.D. degrees in Electrical Engineering from National Chung Hsing University, Taiwan, in 2005 and 2012, respectively. He is also a member of IEEE. His research interests include multimedia, image processing, watermarking, information theory and coding theory.

**Chi-Yuan Lin** received his B.S. degree in Electronic Engineering from National Taiwan University of Science and Technology, Taiwan, in 1988, his M.S. degree in Electronic and Information Engineering from National Yun Lin University of Science and Technology, Taiwan, in 1998, and his Ph.D. degree in Electrical Engineering from National Cheng Kung University, Taiwan, in 2004. Since 1983, he has been with the Department of Electronic Engineering of National Chi-Yi University of Technology in Taiwan where he is now a professor. His research interests include neural networks, multimedia coding, data hiding, and image processing.