IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 18, 2016, accepted October 31, 2016, date of publication November 9, 2016, date of current version March 28, 2017.

Digital Object Identifier 10.1109/ACCESS.2016.2625981

Energy-Efficient Abnormal Nodes Detection and Handlings in Wireless Sensor Networks

FEI LEI[1], LEI YAO[1], DENG ZHAO[2], AND YUCONG DUAN[3]
[1]Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China
[2]School of Information Engineering, China University of Geosciences, Beijing 100083, China
[3]College of Information Science and Technology, Hainan University, Haikou 570228, China

Corresponding author: F. Lei (leifei@bjut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61379126 and Grant 61662021, in part by the Scientific Research Foundation for Returned Scholars, Ministry of Education of China, and in part by the Fundamental Research Funds for the Central Universities, China University of Geosciences, Beijing, China.

ABSTRACT This paper focuses on the abnormal nodes detection of poisonous gas in wireless sensor networks, namely, finding these nodes whose concentrations are higher than the threshold. In order to detect abnormal nodes, we had better collect sensory data from all nodes. However, this strategy requires much more energy consumption, so we should try to wake up these nodes near the abnormal filed. Based on this observation, we propose a novel energy-efficient method to wake them up. The main idea is to let abnormal nodes send out control packets to activate their one-hop neighbor nodes; then, neighbor nodes continue detecting, and finally, all abnormal nodes send information to the sink node through the shortest paths. Thereafter, we further propose to handle these information in the sink node, including extracting boundary nodes, drawing isolines, and estimating the location of leakage source. To extract boundary nodes, we divide all abnormal nodes into different intervals in an ascending or descending order, and then find two nodes with minimum and maximum in each interval, so these nodes are regarded as boundary nodes. As to the second point, we reuse the wide-adopted interpolation methods to draw isolines, such as cubic, nearest, and invdist. Besides, we use interpolation to find the coordinate of the peak, and then, it is deemed to be the leakage source. The experimental results show that our proposed method is feasible.

INDEX TERMS Wireless sensor networks, poisonous gas, energy-efficient, activate, one-hop neighbor node, boundary nodes, isolines, leakage source.

I. INTRODUCTION

Wireless sensor networks (WSNs) have been adopted in many scenarios as a mechanism for dealing with sense-and-send tasks. With the rapid development of sensor nodes in their computational and data collection capacities, these nodes are capable of performing more complicated functions. Definitely, people cannot come close to the area with high concentration when the poisonous gas occurs leakage in a relatively shorter time. If we can find this dangerous area as soon as possible, we can make prompt and proper decision in evacuating and reducing casualties. Generally, a sensor node has a processor, storage unit, transceiver, receiver, battery. Sensor nodes organize themselves to form a multi-hop self-organizing network [1] and jointly perform a task, such as environmental monitoring, object tracking and traffic monitoring [2]. In such tasks, thousands of sensors are deployed in a large-scale interested area to monitor the intrusion or diffusion of specific objects. Most of the existing schemes aimed to track individual targets, e.g., people, animals, enemy vehicles. However, in most cases, it is also indispensable to track the spread range of continuous objects as well, e.g., toxic gases, and radioactive gases. Continuous objects differ from individual targets in that they are continuously distributed across a region. The fundamental characteristic of continuous objects is that they tend to diffuse and have no fixed size and shape over time [3]. Therefore, the traditional technology like GPS can hardly be applicable when monitoring continuous objects, after all, we can't see it or feel it even determine its location at next moment, so we have no choice but to collect information from a large number of nodes to detect and analyze environmental events. WSNs aim to collect data from natural environment, and send these data to the central processing station or sink node.

A paramount challenge of WSNs is the energy efficiency as discussed in [4], which is necessary to prolong the lifetime of network. So we should wake up these nodes near the abnormal filed as much as possible. To address the challenge mentioned above, we propose a novel technique (activating). Current sensor node detection approaches can generally be summarized as three categories: geometric, statistical, and topological methods. The main weakness of statistical method is impractical demand for node deployment, it usually supposes that the deployment of nodes is probability distribution, and then uses some certain properties [5]. So in this paper, we combine geometric method using location information and topological method just collecting information from one-hop neighbor nodes, accordingly, the cost is proportional to the number of one-hop neighbors.

In WSNs, nodes on or near boundary (collectively called boundary nodes) normally play a vital role than other nodes, a region surrounded by abnormal nodes might be equivalent to dangerous zone, finding its boundary nodes can warn us not to close to this zone and help us to do some follow-up work, such as evacuation and rescue. Hough transform [6], [7] or Wavelet transform [8], [9], [10] is a popular tool in boundary detection, but transformation causes extra computational overheads, we only use the coordinate information for simplicity. Intuitively, the boundary of a continuous object depends highly on people's logical decision, hence different people may have different opinions [11]. In addition to extracting boundary nodes, we also do some other work, such as drawing isolines, estimating the location of leakage source. An isoline map displays the layered distribution of the attribute value over the region, drawing isolines help us to get more scientific information. For example, when isolines have a decreasing trend along with surrounding area, we can read out lethal zone, severely injured zone, injured zone, even the approximate zone about gas leakage. These zones convey more detailed information about the dangerous zone. There are several interpolation methods about drawing isolines, including seldom-used way like curve, and frequently-used way like kriging. In real-world applications, finding the leakage source is also a crucial work. For leakage source detection, there are several practices like acoustic wave technology mentioned in [12], but its detection range is too small, and the equipment needs to be used to manually scan in every skeptical region. Hence, this method is time-consuming. Taken these into concern, we put forward a simple method to speculate the location of leakage source based on the basis of drawing isolines. The main contributions of this paper are summarized as follows:

- For the sake of reducing energy consumption, different from the previous algorithms, in our technique sensor nodes that in the sleeping state do not need to consume energy for environment detection, and only wait to be activated by other sensor nodes that are in the active state. Besides, abnormal node sends sensory data packets to the sink node along a multi-hop path.
- For these abnormal nodes in the sink, we make full use of their practical significance, for example, boundary nodes can roughly delineate the dangerous zone; isolines can convey more detailed information about dangerous zone; while leakage source has especially become the key point of remedial work.
- We extract boundary nodes according to their x- and y-coordinates. Besides, we propose a novel method to estimate the location of leakage source, namely interpolation. Finally, in order to demonstrate experimental results, we adopt a visual and intuitionistic way, such as coordinate graph.

This paper is organized as follows. Section II mainly introduces the energy model used in this paper and calculates the energy consumption. Section III presents the sleep mechanisms of abnormal node detection. Section IV shows how to handle these abnormal nodes. Section V-C shows the experimental results of our method. Section VI mainly reviews related work in these research fields, and finally, Section VII concludes this article and future work.

II. PRELIMINARY: CONCEPTS AND ENERGY MODEL

Up to now, countless models have been applied to calculating the energy consumption. In this paper, we reuse the well-adopted first order radio model proposed in [13], whose parameters are presented in Table 1.

TABLE 1. Parameters in the energy model [13].

Name	Description
E_{elec}	Energy consumption constant of the transmit and receiver electronics
ϵ_{amp}	Energy consumption constant of the transmit amplifier
k	The number of bits in one packet
d	The distance of transmission
n	The attenuation index of transmission
r	The communication radius of sensor nodes
$E_{Tx}(k,d)$	The energy consumed to transmit a k bit packet to a distance d
$E_{Rx}(k)$	The energy consumed to receive a k bit packet
$E_{Tx-elec}(k)$	Energy consumption for the transmit electronics
$E_{Rx-elec}(k)$	Energy consumption for the receiver electronics
$E_{Tx-amp}(k,d)$	Energy consumption for the transmit amplifier
$E_{ij}(k)$	Energy consumption for transmitting a k bit packet from a node i to a neighboring node j

Here, the consumed energy for transmitting a k bit packet within a distance d (denoted $E_{Tx}(k,d)$) and receiving a k bit packet (denoted $E_{Rx}(k)$), are given as follows:

$$\begin{aligned} E_{Tx}(k,d) &= E_{Tx-elec}(k) + E_{Tx-amp}(k,d) \\ &= E_{elec} \times k + \epsilon_{amp} \times k \times d^n \quad (1) \\ E_{Rx}(k) &= E_{Rx-elec}(k) = E_{elec} \times k \quad (2) \end{aligned}$$

In total, the energy consumption of transmitting a k bit packet from a node i to a neighboring node j

as $E_{ij}(k) = E_{Tx}(k, d) + E_{Rx}(k)$ is given as follows:

$$E_{ij}(k) = \begin{cases} E_{elec} \times k + \\ \epsilon_{amp} \times k \times d^n & \text{if j is sink node} \\ 2 \times E_{elec} \times k + \\ \epsilon_{amp} \times k \times d^n & \text{otherwise} \end{cases} \tag{3}$$

Note that d is the distance between sensor node i and j (or sink node). The spends on transmitting one packet to another node and to sink node are different, since sink node has no energy constraints and the spend on receiving messages is ignored in this model. The energy required to transmit a message from sensor nodes i to j is the same as that required to transmit a message from sensor nodes j to i, in another word, $E_{ij}(k) = E_{ji}(k)$.

The value of attenuation index of transmission (denoted n in Table 1) is determined by the surrounding environment. If sensor nodes in the network are barrier-free when sending messages, the value of n is set to 2. Otherwise, n is normally set to a value between 3 to 5 when sensor nodes is used for long-distance transmission are distributed in the area of buildings and a vegetation cover [14].

III. ABNORMAL NODES DETECTION

In this section, we will use all our time to introduce how to efficiently detect abnormal nodes. An intuitive idea is to compare the measurement value of sensor node with a predefined threshold; if the value is larger than the threshold, the node actively conducts a detection task around its position [15]. Based on this idea, we propose a novel method about activating. Generally, sensors have their locations (latitude and longitude), which can be obtained through either a GPS device or other localization techniques, for example, ultrasonic positioning. Without loss of generality, a sensor node can be defined through a three-dimensional vector including its *id*, *location* (*latitude* and *longitude*) and *concentration*.

*Definition 1 (**Wireless Sensor Network**):* A wireless sensor network D is a set of sensor nodes. For each sensor node $o \in D$, o is defined as a three-dimensional vector: $(o.id, o.loc, o.c)$, where (i) $o.id$ is the identifier of this node, (ii) $o.loc$ is the geographical position defined by its *latitude* and *longitude* [14], (iii) $o.c$ is the concentration of this node.

A. ASSUMPTIONS

We assume that sensor nodes are of a skewed distribution or uneven distribution in a two-dimensional (2-D) region, and they are dense enough to perform a collaborative sensing. This means that sensor nodes are dense in some sub-regions of the network, while are sparse in other sub-regions. Actually, the scenario of skewed distribution is usual in many real applications, since sub-regions may differ in significance according to the requirement of certain applications [14]. For instance, in the channels of gas transport, there should have dense sensor node deployment near the gas pipeline. In addition to assuming a skewed distribution, we also make three additional assumptions related to the operation in this research. They are:

(i) we consider a static WSN, where the sink node and sensor nodes are fully aware of their own locations; (ii) there is only one leakage source in the center; (iii) for simplicity, neighbor refers to one-hop neighbor node.

B. SENSOR STATES TRANSITION

In terms of power consumption in the network, each sensor node nearly can be in active state or sleep state. When tracking the continuous objects during the process of diffusion, the sensor nodes can enter into sleep state, even if at the present they are sensing the target [16]. Considering such changing situations, we define three kinds of modes for sensor nodes, and active state can be divided into two types:

- *Sleep Mode* (*SM*): In the sleep state, a node does not detect surrounding environment and transmit any messages to others, but it can receive its neighbor messages although its radio is turned off, the processor can still be active, which brings the energy saving.
- *Active-Detect Mode* (*ADM*): When a node receives its neighbor messages, it turns into active state to detect environment but does not transmit messages to sink node if the sensory data is less than the threshold.
- *Active-Send Mode* (*ASM*): After a node turns into active state to detect environment, it needs to report messages to sink node when the sensory data is greater than or equal to the threshold.

Nevertheless, we suppose that head nodes are independent of this state transition and they are always active for responding to dynamic environment [16].

C. NODE DETECTION

The whole framework of node detection is presented in Fig. 1. In the initial stage, deploying unevenly some nodes in a interested region. In order to save energy, the best way is to let few nodes detect surrounding environment without losing accuracy. What's worse, we can't decide which nodes to be preferentially wakened from a sleep state, so we select head nodes being derived from grid cells, which are represented by the *ID*. The main process for network deployment and detection as follows: 1) head nodes cyclically detect concentrations; 2) if the values exceed the threshold, these nodes need to activate their neighbors (if their neighbors have been activated, we need not to activate them again), then neighbors continue to detect and repeat the same work; 3) but if the concentrations of one node and its all active neighbors are normal, the node should turn into sleep mode; 4) nodes with abnormal readings send information to sink node. In the following, we will explain them in detail.

1) THE INITIAL DEPLOYMENT OF NETWORK

This section mainly introduces some deployment work.

(i) Set a region and deploy some nodes

Firstly, we unevenly deploy some nodes in a rectangular region, which is specified with a vector: (*minx*, *miny*, *length*, *width*), where (1) *minx* and *miny* represent the x-coordinate

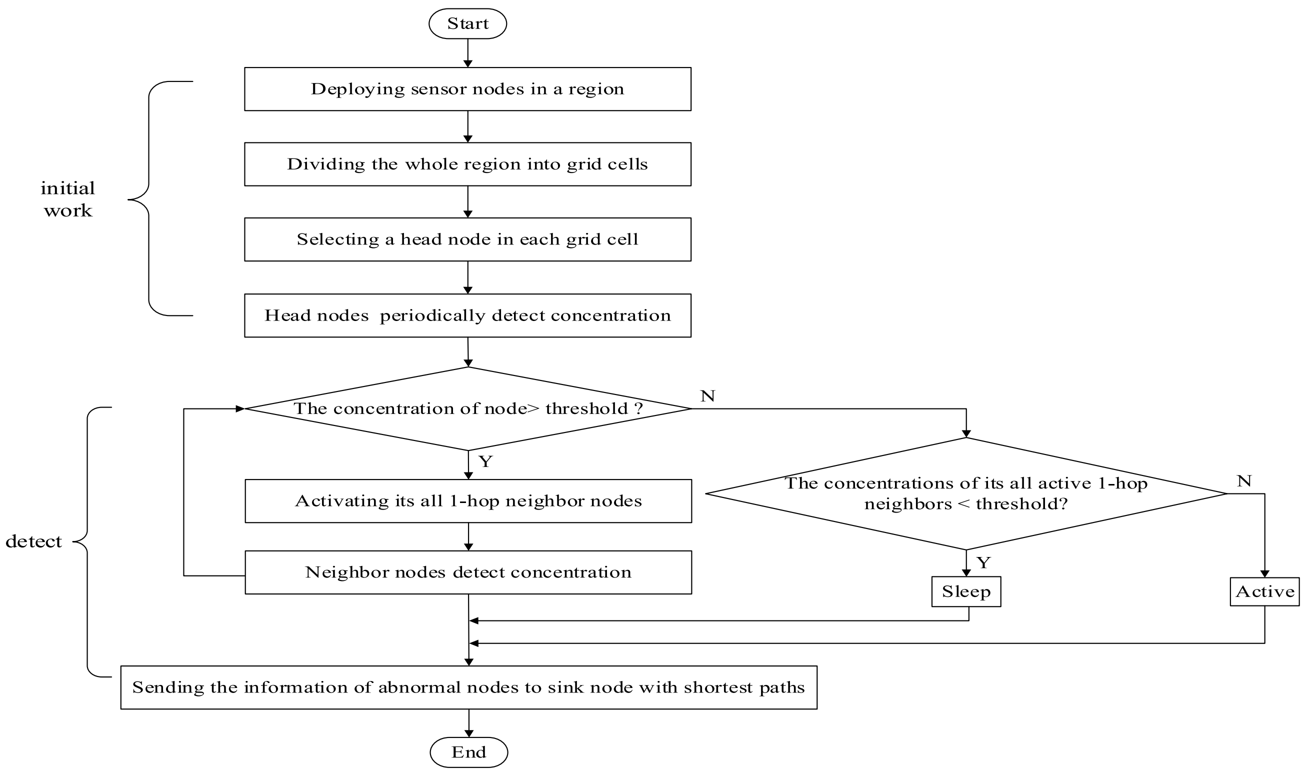

Start
Deploying sensor nodes in a region
Dividing the whole region into grid cells
Selecting a head node in each grid cell
Head nodes periodically detect concentration
initial work
The concentration of node> threshold ?
N
Y
Activating its all 1-hop neighbor nodes
Neighbor nodes detect concentration
The concentrations of its all active 1-hop neighbors < threshold?
N
Y
Sleep
Active
detect
Sending the information of abnormal nodes to sink node with shortest paths
End

FIGURE 1. The whole framework of abnormal node detection including some early work about network deployment.

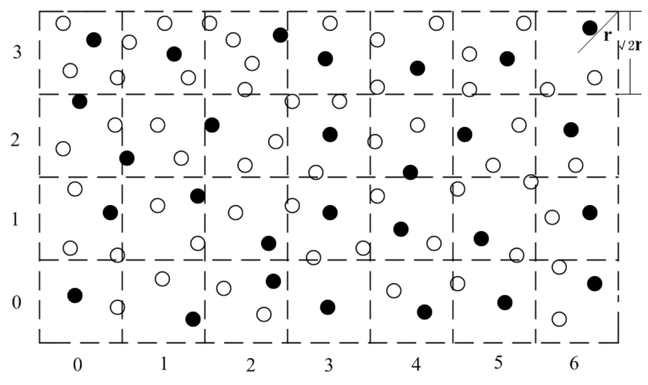


FIGURE 2. Dividing the whole region evenly into grid cells and selecting a head node in each grid cell.

and y-coordinate of the lower left corner of the region, respectively, while (2) *length* and *width* are the length and width of the region. And then allocate id to each node in order to distinguish them.

(ii) Divide the region

Fig. 2 shows an example of the region division denoted by dashed lines, before selecting head nodes, we need to divide the whole region evenly into grid cells, where a grid cell is a square and the side length (denote *gSide*) of each grid cell is $\sqrt{2}$r. In here, *r* represents the communication radius of sensors and assuming that they are same for simplicity. Of course, if the region cannot be divided into equal parts, we should change *minx* and *miny* and increase *length* and *width* so that the region can be divided into equal parts. In other words, the actual area is likely not prescribed in advance. After determining the new length and width, the grid' *ID* that is computed leveraging the value of (1) row (denote *row*) and column (denote *col*) coordinates of grid cells, and (2) the columns of the grid cells (denote *cols*). Generally, $ID = row \times cols + col$, the value of *row* is from 0 to *width* / *gSide* - 1, the value of *col* is from 0 to *width* / *gSide* - 1, and the value of *cols* is *length* / *gSide*. For the purpose of visualization, we use a two-dimensional matrix GridCells[row][col] to represent the grid cells. For example, GridCells[0][0] represents the first grid cell, whose *ID* is 0.

TABLE 2. Coordinate and ID for grid cell shown in Figure 2.

ID	0	...	6	7	...	13	...
Coordinate	(0,0)	...	(0,6)	(1,0)	...	(1,6)	...

The coordinate and ID of grid cells shown in Fig. 2 is described by Table 2, where the first row (i.e., 0) represents the *ID*. The coordinate (i.e., (0, 0)) is shown in the second row of this table.

(iii) Select head nodes

We divide all nodes into different grid cells[i][j] according to coordinate (*i* and *j* are calculated as follows: $i = (o.x - minx)$ / *gSide*, $j = (o.y - miny)$ / *gSide*), then resulting in a few nodes in each grid, as is shown in Fig. 2, different colour is used to mark different variety, there are two kinds of node, which are general node and head node, respectively, and correspond to the white circle, the black circle, respectively. In Fig. 2, each grid cell has its own head node, the head node can be selected according to some methods like the position of sensor node, the energy remaining, and so on. In comparison to those closest to the sink node, we randomly choose one node in each grid.

2) ABNORMAL NODES DETECTION ALGORITHM

This section mainly introduces the algorithm about abnormal node detection. Algorithm 1 details the specific conditions about node when to be activated (line 6-9) and when to enter into sleep state (line 11-13):

```
Algorithm 1 AbnNodeDet
Require: ct: concentration threshold
    c: the concentration of node
    r: the communication radius of sensors
    o: one node
    no: one-hop neighbor nodes
    SN ← sink node
Ensure: AbnormalNodeList: set of abnormal nodes
 1: for each head node do
 2:    periodically detect c
 3:    DetectionNodeList.add(head node)
 4: end for
 5: for each node o in DetectionNodeList do
 6:    if c > ct then
 7:       AbnormalNodeList.add(o)
 8:       activates o.no and DetectionNodeList.addAll(no)
 9:       no continue to detect c
10:    else
11:       if o and its all active no with normal concentration
          then
12:          o is sleep and DetectionNodeList.delete(o)
13:       end if
14:    end if
15: end for
16: send abnormal information in AbnormalNodeList to SN
    through Dijkstra algorithm
17: return abnormal node information
```

(i) Activate neighbor nodes

We let head nodes periodically detect concentration and add them into *detection node list* (denote DetectionNodeList) (line 1 - 4), while, other nodes keep *Sleep Mode* (SM) in order to save energy and extend the sensor lifetime. For each node in DetectionNodeList (line 5), when one node detects abnormal concentration (line 6), it will be added to *abnormal node list* (denote AbnormalNodeList) (line 7)and turn into *Active-Send Mode* (ASM), meanwhile, it activates its all one-hop neighbor nodes, for example, if one node has established its *one-hop neighbor list*, it will broadcast a control packet to its all neighbor nodes in list, then neighbors that receive this message are added to DetectionNodeList (line 8) and turn into *Active-Detect Mode* (ADM) to continue to detect their own concentration and back to line 6 (line 9). Reference [17] mentions that message complexity is $O(n_0)$ because each node only needs to broadcast one message to its 1-hop neighbor nodes when n_0 delegates the number of abnormal nodes. Fig. 3 shows the process of the activation, for instance, one head node detects itself is abnormal then actives all other nodes within the communication range, which is denoted by a dashed circle. After activating, these activated neighbor continue to detect. In Fig. 3, orange circle represent the activated node, the direction of Node A actives Node B is denoted by a arrow from A to B. The process of activation between abnormal head node and its one-hop neighbor nodes is illustrated with bold red arrow, while, the process of activation between other neighbors is illustrated with blue arrow (a arrow from B to C). But if one node detects itself is normal and its all active neighbors are also normal (line 11), this node turns into SM and it has to be deleted from DetectionNodeList (line 12), otherwise it is still active.

(ii) Send abnormal information to sink node

Finally, in order to reduce energy loss, sensor nodes generally use short-range and multi-hop way to send data to sink node. Therefore, we plan to send these abnormal information in AbnormalNodeList to sink node (line 16) with shortest paths using Dijkstra algorithm [18], which is widely recognized as the best method for solving the shortest path between nodes in a graph. For a given source node, the algorithm finds the shortest path between that node and every other. It can also be used for finding the shortest paths from a single node to a single destination node by stopping the algorithm once the shortest path to the destination node has been determined [19]. In here, we firstly adopt to sending each abnormal node' information to corresponding head node in grid, then head node that receives information from abnormal nodes can find a shortest path to send information to sink node, respectively. The main process of algorithm are given as follows: (1) establish a graph, whose edge-weight between two nodes is flagged as the distance of them when they are within a perceived radius of each other, and we assume that

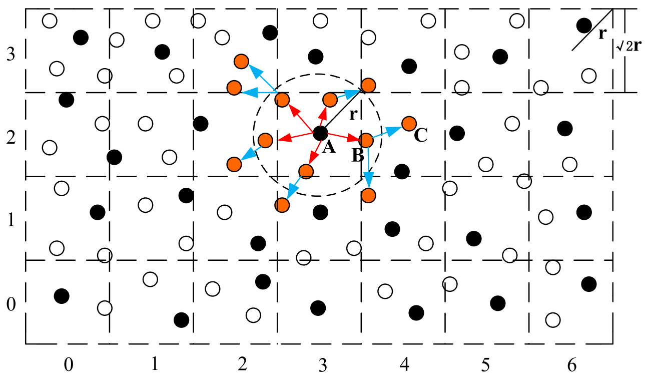


FIGURE 3. Abnormal head nodes activate its one-hop neighbor nodes and neighbors repeat the same work.

perceived radius is $2 \times gSide$, but if the distance exceeds the perceived radius, the weight is set to -1 (we think that this path is blocked); (2) head node that receives the information in each grid is as a source node, while sink node is as a destination node, source node extends to outer layers little by little as a center until it finds the destination node by a shortest path.

(iii) Calculate the energy consumption

By the way, our energy consumption are mainly used to transmitting messages and receiving messages presented in Section II: where (1) abnormal nodes transmit k bit packets and neighbor nodes receive k bit packets; (2) abnormal nodes send information and corresponding head nodes collect information; (3) head nodes send information to sink node but sink node receiving messages is ignored.

IV. THE HANDLINGS OF ABNORMAL NODES

Nodal information is very significant for user, especially abnormal nodes. For abnormal information detected in previous part, we intend to deal with them from the following three aspects.

A. BOUNDARY NODES EXTRACTION

It is worth mentioning that in the real world, event (for example, it usually refers to a kind of abnormal phenomenon like contaminant flows or gas leakage) invariably occurs in a relatively shorter time span, for these emergency scenarios, we are more concerned about the dangerous zone that momentarily signify death or life, finding boundary can warn us not to close to this zone so that boundary nodes are more crucial than other nodes. In a sense, boundary nodes can help to reveal further information about the network topology structure, which is useful for routing and locating. If available, it will drastically reduce the casualties and damage, which would be an indelible contribution for rescue effort.

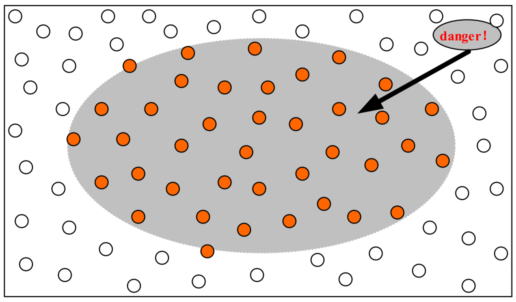


FIGURE 4. A dangerous zone surrounded by abnormal nodes.

Because of the random distribution on the node density, the boundary nodes could only be approximated. No one has clear definition about boundary node, so boundary highly depends on people's logical thinking. In Fig. 4, all orange circles refer to abnormal nodes, the white circles represent normal nodes, and the shaded area represents an dangerous zone. Then our work is to extract boundary nodes like the

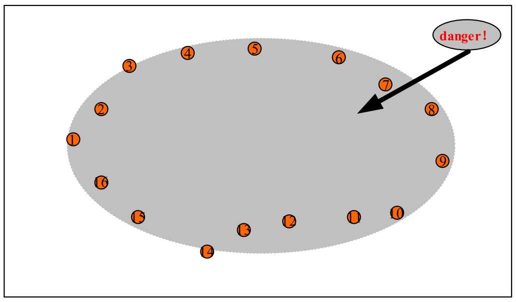


FIGURE 5. The boundary nodes of a dangerous zone.

numbered circles (from 1 to 16) shown in Fig. 5. Detecting nodes on the boundary is similar to *edge detection* in image processing proposed by Abo-Zahhad *et al.* [20] or *boundary estimation* being applicable to medical images proposed by Sofka *et al.* [21].

In the following, we will introduce the algorithm of the boundary node extraction. Actually, an excellent algorithm should find out almost all boundary nodes, but in this paper, we adopt a simple method derived from [22] at the cost of accuracy. The main idea is as follows: 1) in terms of the value of x, divide all abnormal nodes into different intervals by ascending or descending order, then find these nodes with minimal and maximal value about y in the corresponding interval; 2) in terms of the value of y, the idea is same. Its performance depends on the selection of the interval length.

```
Algorithm 2 BouNodeExt
Require: p1 and p2: two step-size parameters
Ensure: boundaryNodes
 1: sort all abnormal nodes in ascending order about X
 2: for every $p1$ numbers do
 3:     use maxYArray C to store X and Y with maximal Y
 4:     use minYArray D to store X and Y with minimal Y
 5: end for
 6: sort all abnormal nodes in ascending order about Y
 7: for every $p2$ numbers do
 8:     use maxXArray E to store X and Y with minimal X
 9:     use minXArray F to store X and Y with maximal X
10: end for
11: combine four arrays C, D, E and F into a resultArray G
12: return resultArray G
```

Algorithm 2 introduces how to extract boundary nodes, in here, we set two step-size parameters $p1$ and $p2$ in advance, which are constant value to divide the number of all abnormal nodes into several parts, but if the number of nodes is aliquant for parameters $p1$ and $p2$, the rest of nodes will be discarded, which brings a certain impact on accuracy. Then we traverse all abnormal nodes to sort them in ascending order about x-coordinate (line 1), after sorting, we use two arrays C and D to store respectively two nodes with maximal y-coordinate and minimal y-coordinate every $p1$ numbers (line 2 - 5).

For y-coordinate, we also use the same method to sort all abnormal nodes in ascending order (line 6) and find the two nodes with minimal x-coordinate and maximal x-coordinate every *p*2 numbers and store them respectively in two other arrays E and F (line 7 - 10). Finally, we output all nodes that we store them in four arrays to an array G (line 11). And according to common sense, the more smaller the step parameter is, the better the result is, but meanwhile, many inner nodes are ineluctably extracted.

B. ISOLINES DRAWING

We come up with drawing isolines so as not to waste a lot of information about inner nodes, as we all know, isolines drawing has been widely used in many sensor fields. A isoline map of an attribute (e.g., height, pressure, temperature, concentration) displays the layered distribution of the attribute value over the region. It often consists of a set of regions outlined by different values. In this paper, we only research the concentration isogram of poisonous gas. The distribution and the influence of the gas can be visualized, by a visual graphics, we get more scientific and detailed information, for example, we can read out lethal zone, severely injured zone, injured zone, even the approximate area of leakage source.

It is very difficult to achieve by individual node. A naive approach is to collect sensory data from all the sensor nodes, obviously, sending a huge amount of nodal information to the sink incurs heavy traffic. To remedy this problem, we firstly find these abnormal nodes detected in Section III and send to sink, and then construct the isoline map at the sink. The main idea of drawing isolines is that knowing some discrete points, then by interpolation, approximately getting a three-dimensional surface Z = f(x, y) through these discrete points, then sectioning the surface with a plane of Z = C, thereafter, projecting to the x-y plane to obtain a concentration isoline of C. Fig. 9 or Fig. 10 presents a graphic model of drawing isolines, in figure, green filled points represent the interpolation points, black filled points represent the known points. The surface is through these green points, we can see that these black points are approximative on this surface. The lines on x-y plane represents some isolines. There is no doubt that if the smaller the interval of the interpolation, the more the number of black points is on surface, at the same time, the greater calculation is required.

The core part of this process is interpolation. In Matlab, interp3 and griddata are both two common interpolation functions, the difference between them are: (i) the interpolation function about interp3 must require that known data points form regular matrix, or so-called grid; but the known data points of griddata function does not be required to be regular arrangement, especially for some random data taken from the experiment, it will generate good result; (ii) interp3 is only suited to internal interpolation, while, griddata can calculate internal and external interpolation. Based on an overall consideration of various factors, we choose griddata function. The griddata function calls format on the interpolation as follows: Z = griddata (x, y, z, X, Y, method), in here,

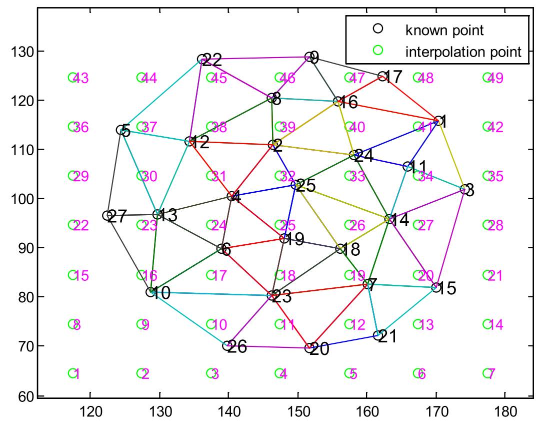


FIGURE 6. A example of cubic about its main working principle.

(x, y) is the coordinate of known discrete point; z is the concentration of known discrete point; (X,Y) is the coordinate of interpolation point generated by meshgrid function; Z is the concentration of interpolation point; method refers to interpolation method, including linear, cubic, nearest, v4 and invdist. By the way, the meshgrid function calls format as follows: [X, Y] = meshgrid (min (x): 10: max (x), min(y): 10: max (y)), number 10 represents interval between two points on the same level, of course, we can also set other numbers, such as 2, 5 and so on, for example, Fig. 6 or Fig. 7 shows a result about meshgrid function when the two green points' interval is set to 10. (min (x), max (x), min (y), max (y)) refers to the control area of known discrete points, we can also increase it a little, such as (min (x) - 5: 10: max (x) + 5, min (y) - 5: 10: max (y) + 5), it increases the height and width of the control area towards to two sides about 5m.

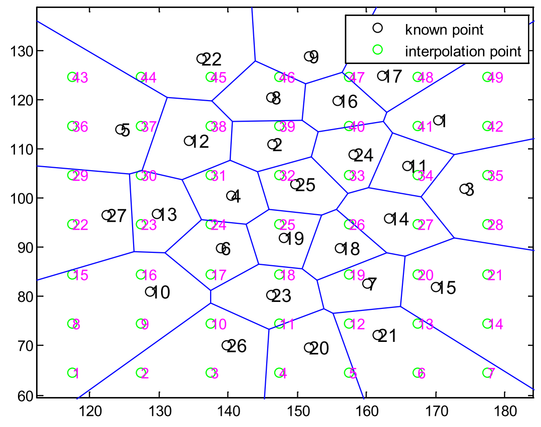


FIGURE 7. A example of nearest about its main working principle.

Then we choose three methods among this five interpolation methods, namely cubic, nearest and invdist. After interpolation, we start to drawing three-dimensional surface and isolines, their call formats are meshc (X, Y, Z) and contour (X, Y, Z), respectively, but in order to clearly show the result, we all tend to draw isolines with filling effect, namely contourf (X, Y, Z). The following parts focus on three interpolation methods' working principles:

- *Cubic* [23]: Cubic interpolation method is based on the cubic equation of Delaunay, it consists of the following four steps: (1) According to discrete ordinates, referencing convex hull algorithm to obtain the convex hull of discrete points; (2) Constructing discrete point's Delaunay triangulation referenced in [24], which maximizes the minimum angle of all the angles of the triangles in the triangulation and requires that the circumcircles of all triangles have empty interiors; (3) According to the grid method to obtain the coordinates of grid points (or interpolation points), there are two grid method, one is setting the interval of vertical and horizontal coordinates via the control range of discrete points, for example, [X, Y] = meshgrid (min (x): 10: max (x), min (y): 10: max (y)), another is setting the number of point, for example, X = linspace (min (x), max (x), 20), Y = linspace (min (y), max (y), 20), it refers to generate twenty numbers between the control range of x- or y-coordinate, as mentioned above, we use the first one; (4) Because it requires that points within Delaunay triangle, so traversing all interpolation points in the region, according to three known discrete points of the triangle surround each interpolation point to calculate the concentration value of this interpolation point using cubic equation.
  Fig. 6 shows the main working principle of cubic, in Fig. 6, green hollow points numbered by pink number from 1 to 49 represent the interpolation nodes, black hollow points numbered by black number from 1 to 27 represent the known discrete nodes. Black points construct a Delaunay triangulation, taking black points 10, 23, and 26 as the example, because they are not in a same straight line, so they can uniquely determine a plane Ax + By + Cz + D = 0, then working out the cubic equation via substitution method, we can calculate this four coefficients A, B, C and D. Finally, we obtain the value of z-coordinate of green point 10 within Delaunay triangle generated by the three black points 10, 23 and 26 because of knowing green point' x-coordinate and y-coordinate. However, this method causes a problem that is appearing NaN (Not a Number) as shown in Fig. 15(a), because some green points are located out of Delaunay triangulation, such as green points 1, 2, 3, 4 and so on.
- *Nearest* [23]: Nearest neighbor interpolation method is also known as Thiessen polygon method, Thiessen polygon (or Dirichlet polygon or Voronoi diagram [25]) proposed by Dutch meteorologist A.H.Thiessen is an analytical method, and the dual graph for a Voronoi diagram corresponds to the Delaunay triangulation for the same set of points, that is say, these points are connected into a triangular network, and then a perpendicular bisector of each side is the division that we desire. Originally, it used to calculate average rainfall via rainfall data derived from some discrete weather station, but now in geographic analysis and GIS, Thiessen polygon is often used for quick assignment. In fact, an implicit assumption about the nearest neighbor interpolation method is any grid point taking the attribute value from its nearest point.
  Fig. 7 shows the main working principle of nearest, similarly, in Fig. 7, green hollow points represent the interpolation nodes, black hollow points represent the known discrete nodes. Black points generate Thiessen polygons, in each Thiessen polygon, there is a black point, the attribute values of other green points take the attribute value of this black point, taking black point 26 as the example, the concentration values of green points 2, 3, 10 are both equal to the concentration value of black point 26. However, this method causes a problem, namely surface is discontinuous and not smooth as shown in Fig. 16(b), because the function value of interpolation point takes the value of the known nearest point.

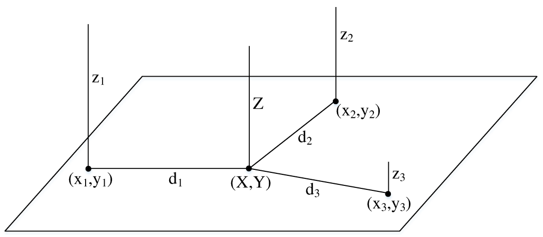


FIGURE 8. A example of invdist about its main working principle.

- *Invdist*: Inverse Distance Weighting (IDW) [26] is a type of deterministic method for multivariate interpolation with a set of known scattered points. The values of unknown points (or interpolation points) are calculated with a weighted average of the values available at the known points. The name given to this methods is motivated by the weighted average, since it resorts to the inverse of the distance to each known point when calculating weights. A general form of calculating the Z at point (X, Y) based on samples z_i (i = 1, 2, ..., N^*) using IDW is:

$$Z(X,Y) = \begin{cases} \frac{\sum_{i=1}^{N^*} w_i \times z_i}{\sum_{i=1}^{N^*} w_i} & \text{if } d_i \neq 0 \text{ for all i} \\ z_i & \text{if } d_i = 0 \text{ for all i} \end{cases} \quad (4)$$

  Where $d_i = d((X, Y), (x_i, y_i))$ is distance from the unknown point (X, Y) to the known point (x_i, y_i). And $w_i = \frac{1}{d_i{}^p}$ is a simple IDW weighting function and p is a positive real number, called the power exponent. Generally, p is set to 2. Besides, z_i is the attribute value of an known discrete point (x_i, y_i). N^* is the total number of known points.
  Fig. 8 helps to understand the main working principle of invdist, (x_1, y_1), (x_2, y_2) and (x_3, y_3) denote three known points, (X, Y) denotes an unknown point,

z_i (i = 1, 2, 3) is the attribute value of known point, and Z is the attribute value of unknown point. Firstly, calculating the distance d_i between unknown point and each known point, respectively; secondly, calculating the weight w_i of each known point; finally, calculating the attribute value Z at point (X, Y) as follows: $Z = (\frac{1}{{d_1}^2} \times z_1 + \frac{1}{{d_2}^2} \times z_2 + \frac{1}{{d_3}^2} \times z_3)/(\frac{1}{{d_1}^2} + \frac{1}{{d_2}^2} + \frac{1}{{d_3}^2})$.

C. ESTIMATION FOR LEAKAGE SOURCE

Leakage of dangerous gas causes great harm for human life, once the leakage happened, if we can not promptly find and stop it, which not only causes resource waste, economic loss and environmental pollution, but also endangers personal safety, and even causes heavier disaster accident. On the contrary, if finding timely leakage source location, it not only saves lives, but also provides a reliable basis for the remedial work, so that it saves a lot of manpower and resources.

How to quickly and accurately locate the source of leakage has becoming a hot and difficult question on positioning technology. Since the diffusion of gas has generally a hidden and unexpected event and we usually can not know the geographical location of leakage source in advance. But we can obtain the concentration of gas in the environment through the sensor, and speculate that the information of leakage source. Due to random deployment, in most case, it maybe has some sensors surrounding the leakage source instead of locating in there, so we only can approximately estimate the location of source. Drawing isolines can obtain the approximate area of gas leakage, but it is not the leakage source, there are many method applied to find the leakage source, such as mobile robot [27], but the robot can not reach some dangerous zones, or places with obstacles, motivated by drawing isolines, we propose a new method, namely interpolation. Because of by interpolation mentioned in Section IV-B, known discrete points and interpolation points have their own attribute value (concentration), finding the maximal concentration value between known discrete points and interpolation points, its corresponding coordinate can be considered as the position we are looking for. In this section, we similarly adopt these three interpolation methods and the principles of algorithms are not discussed again.

In fact, the peak has only two situations. Fig. 9 and Fig. 10 show the possible results about maximal concentration value denoted by bold red point. In figure, green filled points represent the interpolation points, while black filled points represent the known points. As we have seen, in Fig. 9, if the interpolation interval is too large to having an influence on maximal value, so that the peak is from known points. But in Fig. 10, with the decrease of interval, interpolation will gradually play a leading role, so that the peak is from interpolation points.

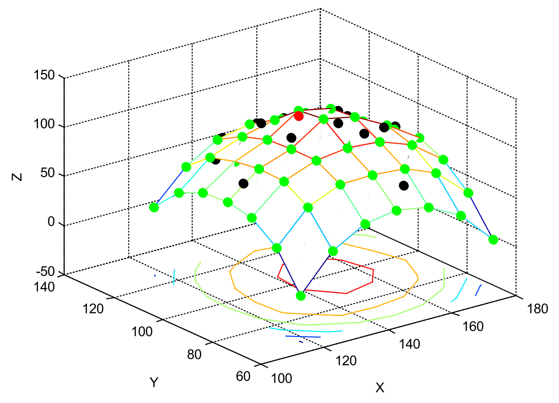


FIGURE 9. The peak is from known points when the interval of interpolation is large.

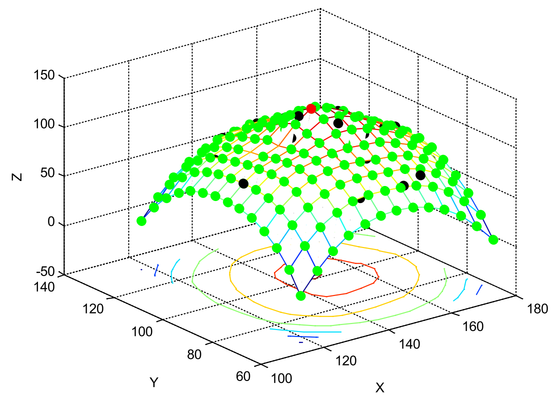


FIGURE 10. The peak is from interpolation points when the interval of interpolation is small.

V. IMPLEMENTATION AND EVALUATION

The prototype has been implemented in Java program (Section III) and Matlab 7.11.0 (R2010b) program (Section IV) on a desktop with Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz, 4GB memory and 32-bit Windows system. In the following, we will introduce the prototype implementation and present our experimental results.

A. IMPLEMENTATION

The experiments has been implemented for evaluating the method developed in previous sections with (i) various numbers of sensor node and (ii) different skewness degree.

We generate six sets with the different number of sensor nodes: 1, 500, 2, 000 and 2, 500. Sensor nodes are distributed unevenly in the network region with different skewness degrees: 10%, 20% and 30%. A skewness degree (denote sd) is computed using the formula: $sd = \frac{dn - sn}{N}$, where dn and sn are the number of sensor nodes in *dense* sub-regions and *sparse* sub-regions, and N is the number of sensor nodes in the network region including *dense* and *sparse* sub-regions (i.e., $N = dn + sn$). In order to form the skewness distribution, in this experiment, we divide the network region evenly into four sub-regions. A pair of sub-regions in the diagonal is regarded as dense sub-regions, while the another pair of sub-regions is regarded as sparse sub-regions. We deploy more sensor nodes in dense sub-regions, while less sensor nodes in sparse sub-regions. For instance, if a WSN contains N (for

TABLE 3. Simulation parameters setting.

Parameter Name	Value
Area size	300m × 300m
Minx and miny	0m
Number of sensor nodes	1,500 to 2,500
Skewness degree	10% to 30%
Communication radius	50m
Number of bits in one packet(k)	1
Attenuation index of transmission(n)	2
Energy consumption constants of the transmit and receiver electronics(E_{elec})	$50nJ/bit$
Energy consumption constant for the transmit amplifier(ϵ_{amp})	$0.1nJ/(bit \times m^2)$
The threshold of concentration	70mg/m^2
Time interval for head node detection	10 seconds

instance, 1,000) sensor nodes, and the skewness degree is set to sd (for instance, 60%), then, $N \times (1 - sd)$ (for instance, $1000 \times (1 - 60\%) = 400$) sensor nodes are deployed evenly in the network region firstly, and the remaining $N \times sd$ (for instance, $1000 \times 60\% = 600$) sensor nodes are distributed to the two dense sub-regions in a random manner. Therefore, a WSN with N sensor nodes distributed in a skewness degree of sd is constructed [14]. The dense sub-region and sparse sub-region are set to the same in size.

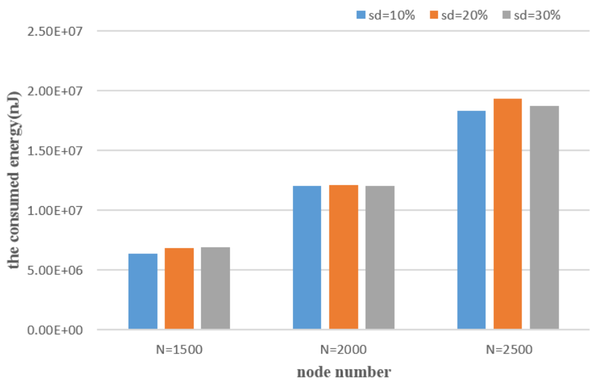


FIGURE 11. The consumed energy of abnormal node detection.

B. PARAMETERS SETTING

Once the nodes are determined, an appropriate radio transmission range is chosen according to nodal density, such that the network is connected. Each node connects to its neighbors within its radio transmission range [28]. In our simulated experiments, we assume that two sensor nodes can communicate with each other when they are within a distance of the communication radius $r = 50$ m, and thus the side length of grid cells is set approximately to 70m. According to the energy model presented in Section II, we set $E_{elec} = 50$, $\epsilon_{amp} = 0.1$, $n = 2$, and the packet transmitted to sensor nodes or the sink node is set to one unit of bit. The unit of energy is $nJ(nJoule)$, which is equal to $10^{-9}J(Joule)$. Besides, we assume that when gas concentration reaches 70mg/m^2, it is harmful to our body. The whole area size is $length \times width$ (for instance, $300m \times 200m$) and the x-coordinate and y-coordinate of the lower left corner of the region are $minx$ (for instance, $0m$) and $miny$ (for instance, $0m$). The time interval for head node detection is 10 seconds. In order to simulate the process of gas leakage, we use a formula to set each nodal concentration value C_i at the point (x_i, y_i):

$$C_i = (1 - dis((x_i, y_i), center)/dis_{max}) \times C_{max} + F \quad (5)$$

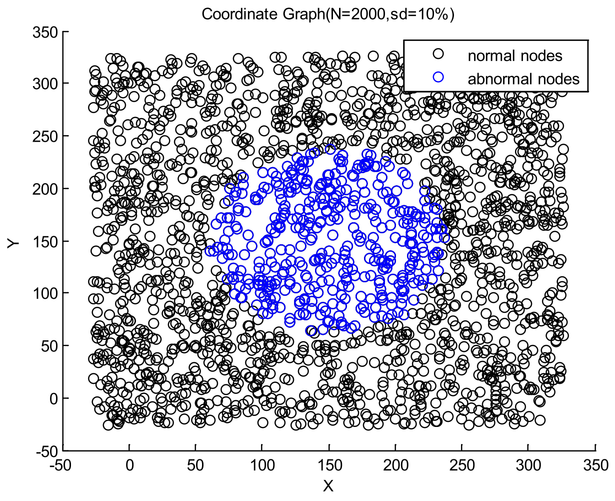


FIGURE 12. The coordinate graph of normal and abnormal nodes.

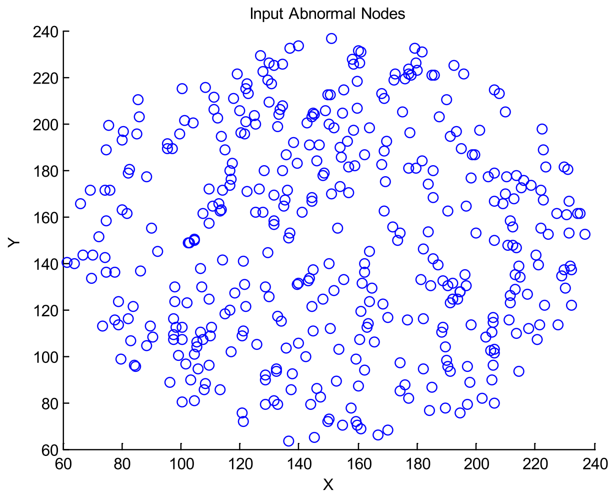


FIGURE 13. The coordinate graph of inputting abnormal nodes.

In the formula, $dis((x_i, y_i), center)$ delegates the distance between the point (x_i, y_i) and a virtual node located at center whose coordinate is $(minx + \frac{length}{2}, miny + \frac{width}{2})$ (for instance, (150, 150)), and dis_{max} delegates the distance between the farthest node and center node, while, C_{max} delegates a maximum concentration value, which is set to 110mg/m^2 according to the practical need, F is a little random number about fluctuation ranging from -1 to 1. From this formula, we can find that the concentration values of nodes near to the center are larger, then it gradually decreases along with the increase of the distance. The settings of parameters are summarized in Table 3.

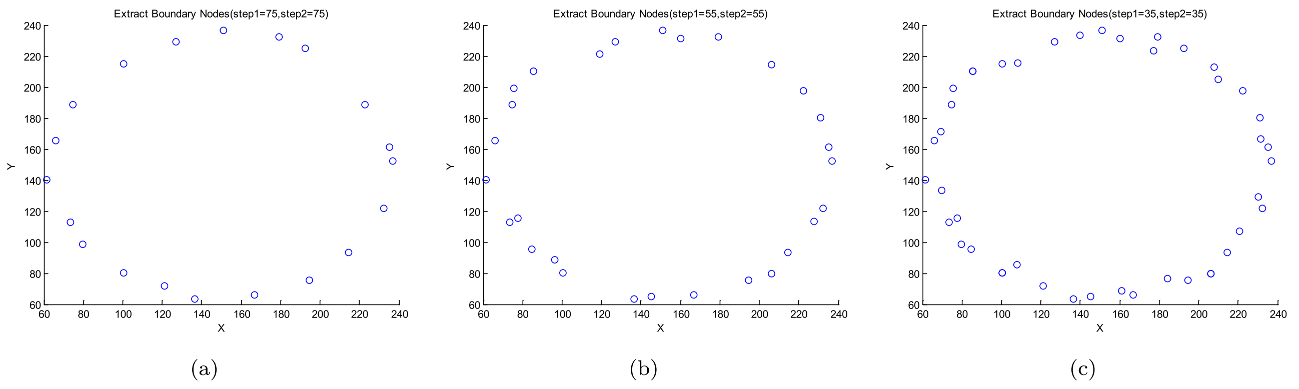


FIGURE 14. **The coordinate graph of outputting boundary nodes under different values about step-size parameter. (a) 75. (b) 55. (c) 35.**

C. *EXPERIMENTAL RESULT*

1) DETECT ABNORMAL NODES

This section shows the experimental results of Section III. Fig. 11 shows the consumed energy comparisons of node detection under various sensor node sets and different skewness degrees in the form of bar graph. The number of sensor nodes varies from 1, 500 to 2, 500, the skewness degree varies from 10% to 30% and and correspond to the blue, orange and grey rectangle, respectively in figure. This figure shows that, the bigger the node number is, the more the consumed energy is required. But given the same number of sensor nodes, we can find that there is little difference between the three skewness degrees. From this figure, we know that the consumed energy approximately varies between $5 \times 10^6 (nJ)$ and $2 \times 10^7 (nJ)$.

Taking $N = 2,000$ and $sd = 10\%$ as the example, Fig. 12 shows the result of node detection in the form of coordinate graph. In Fig. 12, black hollow points denote normal nodes, while, blue hollow points denote abnormal nodes. The horizontal direction is x-coordinate ranging from -50m to 350m and the vertical direction is y-coordinate ranging from -50m to 350m because of changing *minx* and *miny* stated in Section III-C.1. Their intervals are both 50m.

2) EXTRACT BOUNDARY NODES

This section shows the experimental result of Section IV-A. We set two step-size parameters $p1 = p2$. Similarly, taking $N = 2,000$ and $sd = 10\%$ as the example, it's worth noting that blue hollow points in Fig. 13 are also blue hollow points in Fig. 12. At first sight, because the ranges of coordinate in horizontal and vertical direction are both from 60m to 240m and the intervals are both 20m, so Fig. 13 has a magnified effect compared to the blue points in Fig. 12. Fig. 14 shows the result of extracting boundary points from blue hollow points in Fig. 13 or in Fig. 12, and Fig. 14(a) - (c) tell us that the effect of extraction becomes better and better with the decrease of the values of two step-size parameters, for example, the values are set to 75, 55 and 35, respectively.

3) DRAW ISOLINES

This section shows the experimental results of Section IV-B. The interpolation interval is set to 20. From three aspects, we compare and analyze three interpolation methods, which are cubic, nearest and invdist.

(i) Firstly, in terms of numerical value, Fig. 15(a), Fig. 15(b) and Fig. 15(c) label the concentration values to the corresponding coordinates, and green hollow points numbered by pink number denote unknown interpolation nodes. Fig. 15(a) tells us that interpolation method using cubic generates a large number of NaNs, which is very limited in the practical application; Fig. 15(b) tells us that interpolation method using nearest generates some same attribute values in a small region. Their shortcoming are both marked in red oval. Comparatively speaking, Fig. 15(c) has neither a NaN nor same attribute value.

(ii) Secondly, in terms of three-dimensional surface, Fig. 16(a), Fig. 16(b) and Fig. 16(c) fit these green points to a surface, black filled points denote known abnormal nodes, green filled points denote unknown interpolation nodes. Fig. 16(a) shows that surface is incomplete when adopting cubic-based method; Fig. 16(b) shows that surface is not smooth and discontinuous when adopting nearest-based method; Comparatively speaking, Fig. 16(c) using invdist-based method is complete and continuous.

(iii) Finally, Fig. 17(a), Fig. 17(b) and Fig. 17(c) show the results of drawing isolines under different methods. These figures show that, each point is equal value in same isoline (such as 100) and two adjacent isolines whose interval is 5 filled with different colours can be decreasing trend according to the surrounding area, different values represent different degrees of severity and the region with greater value is naturally more dangerous, we even can speculate that leakage region is located in the wine red area. Due to the number of point is too little at isoline 105, so that Fig. 17(a), Fig. 17(b) cannot be presented. Fig. 17(c) maybe conforms to the actual concentration setting in experiment when compared with Fig. 17(a) and Fig. 17(b). Because the outer circle of Fig. 17(a)

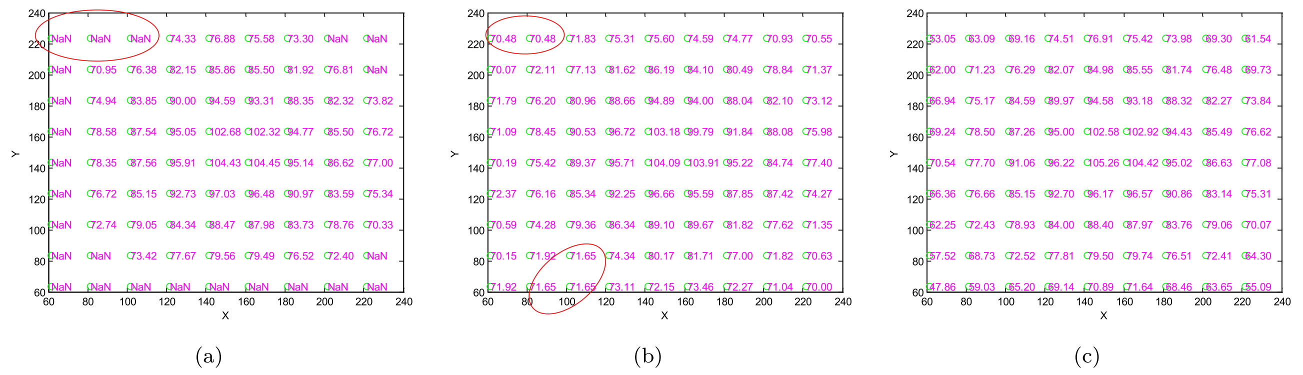


FIGURE 15. Comparing and analyzing three interpolation methods from the angle of numerical value. (a) Cubic. (b) Nearest. (c) Invdist.

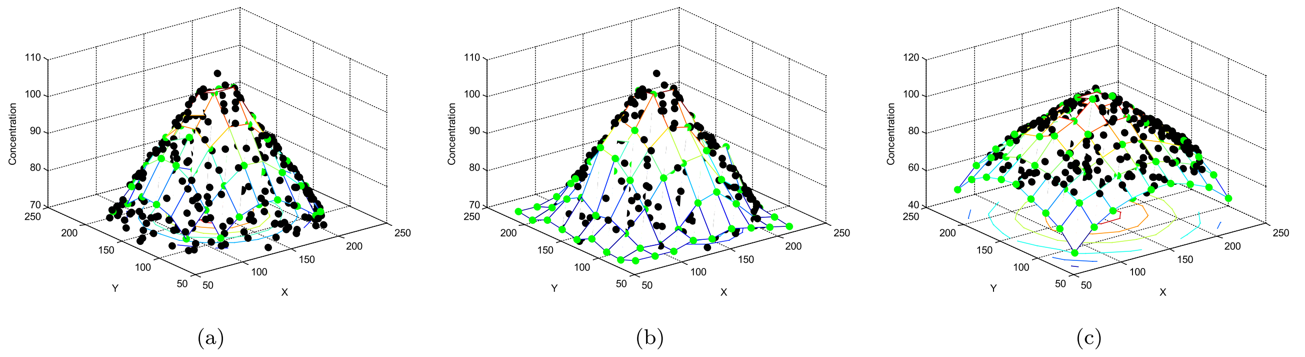


FIGURE 16. Comparing and analyzing three interpolation methods from the angle of surface. (a) Cubic. (b) Nearest. (c) Invdist.

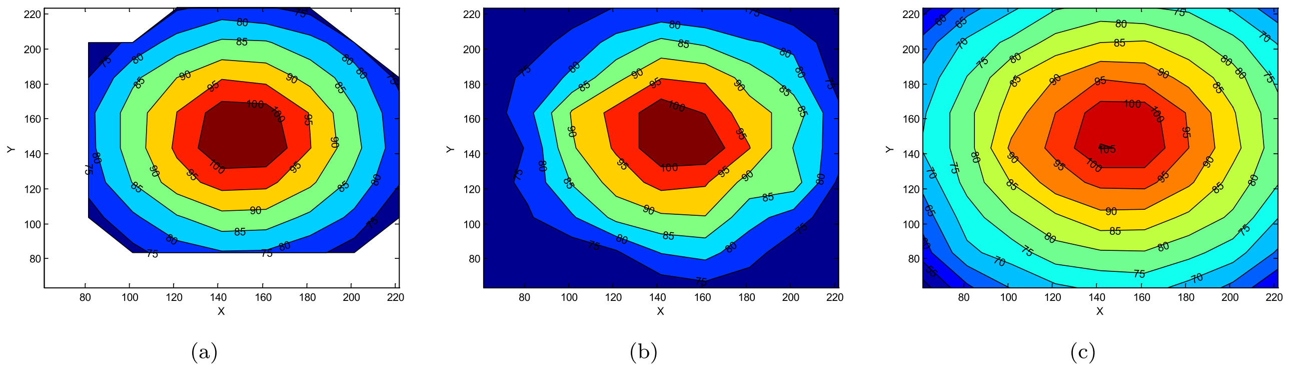


FIGURE 17. Comparing and analyzing three interpolation methods from the angle of isoline. (a) Cubic. (b) Nearest. (c) Invdist.

TABLE 4. The result of estimation shown in Figure 18 – 20.

Interpolation	Interval	The Coordinate of Peak
Cubic	20	(153.2205, 155.0034)
	10	(153.2205, 155.0034)
	5	(151.4779, 153.5423)
	2	(152.4779, 152.5423)
Nearest	20	(153.2205, 155.0034)
	10	(153.2205, 155.0034)
	5	(153.2205, 155.0034)
	2	(153.2205, 155.0034)
Invdist	20	(153.2205, 155.0034)
	10	(153.2205, 155.0034)
	5	(151.4779, 148.5423)
	2	(150.4779, 150.5423)

doesn't get shown and in Fig. 17(b), the effect of distortion is obvious, especially, in the final.

4) ESTIMATE THE LEAKAGE SOURCE

This section shows the partial results of Section IV-C. Similarly, we adopt three interpolation methods. The interval of interpolation is set to 20, 10, 5, 2, respectively. Because there is no difference when interval is set to 20 and 10, so the results of 10 is ignored. In each figure, black filled points denote known abnormal nodes, green filled points denote unknown interpolation nodes, and red filled point denotes the peak between black points and green points. (153.2205, 155.0034, 107.7836) is the maximal point among all known points.

(i) Cubic

Fig. 18(a), Fig. 18(b) and Fig. 18(c) are the experimental results about cubic-based method. In Fig. 18(a), because the interpolation interval is too big enough to playing a role

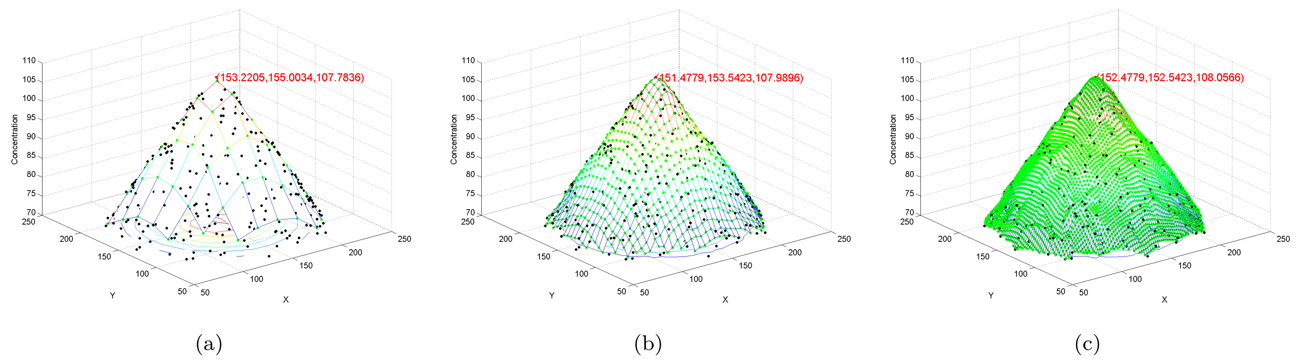


FIGURE 18. Estimating the location of leakage source using cubic-based method under different interpolation intervals. (a) 20. (b) 5. (c) 2.

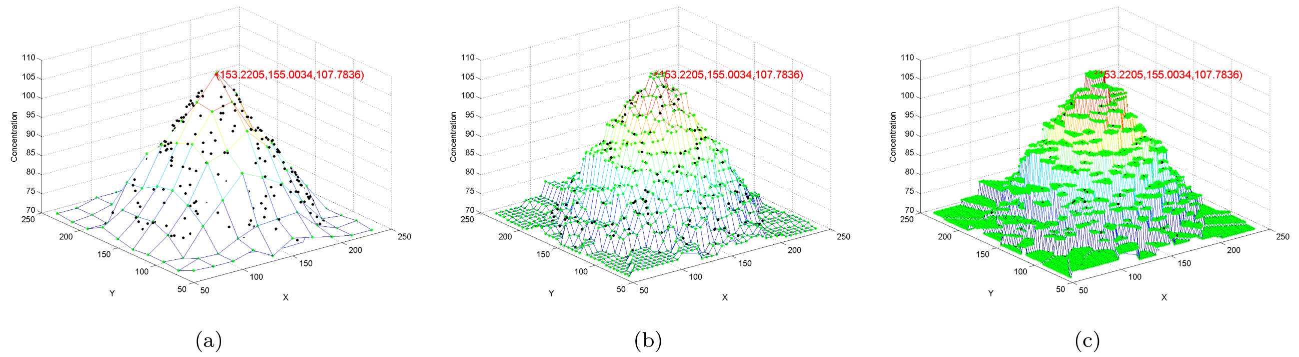


FIGURE 19. Estimating the location of leakage source using nearest-based method under different interpolation intervals. (a) 20. (b) 5. (c) 2.

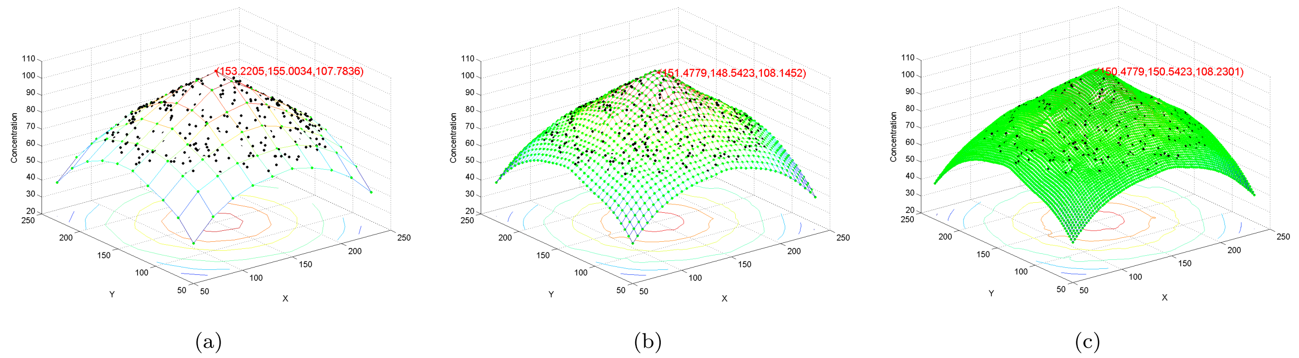


FIGURE 20. Estimating the location of leakage source using invdist-based method under different interpolation intervals. (a) 20. (b) 5. (c) 2.

in estimating the location of leakage source, so the peak is from these known points, then interpolation gradually starts to work as shown in Fig. 18(b) and Fig. 18(c), so the peak is from interpolation points.

(ii) Nearest

Fig. 19(a), Fig. 19(b) and Fig. 19(c) are the experimental results about nearest-based method. Because of the intrinsic characteristic (namely nearest neighbor) of this algorithm, therefore, the peak can only take the value of the known maximal point.

(iii) Invdist

Fig. 20(a), Fig. 20(b) and Fig. 20(c) are the experimental results about invdist-based method. The whole changing trend of these three figures is very similar to the first interpolation method. Delightfully, given the same interval, such as 5 or 2, it is closer to the actual leakage source (150, 150).

In addition to these above pictures, we also record the coordinate of peak (x_m, y_m) in the form of a table. By the way, z-coordinate is concentration instead of the height that we often say, so we don't consider it when recording the experimental result described in Table 4, where the first column represents three interpolation methods. The interval of interpolation is shown in the second column of this table. The third column is the coordinate of peak. In fact, it's just another expression taking the place of picture. For example, the first row can be interpreted as: when the interpolation method and interval are cubic and 20, respectively, the coordinate of peak is (153.2205, 155.0034) and corresponds to the Fig. 18(a).

In order to facilitate observation, Fig. 21 demonstrates a line chart about their distance between peak and center under different methods and intervals. In this experiment, the coordinate of center is (150, 150), so their distance can be computed leveraging a formula as follows: $dis(peak, center) = \sqrt{(x_m - 150)^2 + (y_m - 150)^2}$. It is obvious that the smaller the distance is, the higher the accuracy is. In figure, blue line represents a nearest-based method, orange line represents a cubic-based method, and grey line represents a invdist-based method. Fig. 21 shows that, when interpolation method uses nearest, the distance never changes because its peak can always come from the known points, for other two methods, at the beginning, because of larger interpolation interval, such as 20 or 10, two interpolation methods don't play a role, afterwards, the distance is shrinking with the decrease of interval, such as 5 or 2, but it is quite clear that the invdist-based method has a higher accuracy.

VI. RELATED WORK

A. THE DETECTION OF NODE

Undoubtedly, deciding which nodes to be awake preferentially is still a long-standing and challenging task, in addition to taking the energy into consideration, we also have to achieve higher coverage. The request for efficient node detection has led to a lot of research thrusts. There are essentially two solutions to the problem of conserving energy in sensor networks.

1) PERIODICAL DETECTION

Under normal circumstances, we naturally think of detecting periodically. For example, [29] mentions a node periodically changes from active to listening mode. If there is no reading difference between the node and its neighbors over a time, then a node will periodically changes from listening to sleeping mode after a predefined time. And the idea of MAC protocols based on preamble sampling in [30] has done similar research mentioning every node toggles periodically the radio from sleep to active mode to detect incoming packets. If during the sampling, there are no packets to be received, the node toggles the radio to sleep. Otherwise, the node keeps the radio on for the time necessary to receive the incoming packets, and then it turns the radio off. Besides, a node in [31] periodically broadcasts HELLO message to detect the change of its neighbors, the changing of a neighbor set are due to node failure, node movement, and so on. When a node joins or exits the network, the network will be locally reconstructed. Motivated by [30], reference [2] also proposes to each node sleeps for some time and then wakes up to sense the target and listens to check whether its neighbors want to communicate with it. Obviously, them can't save energy because of all nodes to be involved in this work. To remedy this problem, we only select few representative nodes (e.g., head nodes) to detect periodically.

2) SELECTIVE WAKE-UP METHOD

Due to the random deployment of sensor nodes, each node's coverage area is very likely to overlap with others [32]. For the redundant node problem in a large scale sensor network, Voronoi Diagram itself is a popular method. Based on Voronoi Diagram, which triangulates the sensing field into different regions around each node, [33] designed a new scheduling scheme, which can decide the nodes to be on or off and maintain the original coverage area. This scheme mainly take a node out of service temporally in order to reduce energy consumption, but constructing Voronoi Diagram is not an easy work. At the beginning, we come up with determining whether the nodes are redundant based on Voronoi Diagram, then we decide to quit because of heavy workload and the lack of professional knowledge. Reference [34] also proposes a solution to determine and turn off redundant nodes. Firstly, this paper classifies nodes by the different locations of the neighbor nodes, and then studies the constraint relations between coverage of neighbor nodes and working nodes in each group. Finally, according to different redundancy rules, it determines to turns off the redundant nodes. But because of the different redundancy rules, the results vary significantly. The authors of [35] use the redundant nodes in dense sensor networks and dividing them into different groups to be scheduled by turns can effectively extend the network lifetime. Motivated by [35], we refer to the idea of grid-based.

In a nutshell, in this paper, our work can be mainly summarized into one sentence: Selecting a head node in each grid and only let them detect periodically, and the rest of nodes are activated by others. Head node selection is not a new technology, a cluster head node consume much more energy than a non-cluster head node, since they are required to consume much more energy when aggregating sensory data from other sensor nodes in the cluster [14]. In order to balance the energy load among all the nodes in each grid cell, [36] was one of the latest papers using LEACH method, which electing new cluster head inside the cluster when the resident energy of the former cluster head is lower than standard. In here, we randomly choose one node in each grid cell for convenience as it is not the core content of this paper.

B. THE HANDLINGS OF NODE

1) BOUNDARY NODES EXTRACTION

Nowadays, considerable efforts have been applied to boundary node extraction, [37] proposes a knowledgeable method based on Hough transform, this transformation projects some non-local phenomenons to some points in the dual-space and the locations of the nodes to some lines that separates the dual-space. Reference [38] proposes an universal hierarchical architecture based on Wavelet transform, this transformation is usually used for multi-resolution analysis(MRA). Motivated by [38], Nowak and Mitra *et al.* [39] proposes a novel multi-hierarchical method to identify boundary nodes by trimming some uncorrelated portions. In [40], three methods are proposed, including statistical thresholding, digital image processing and pattern recognition. The statistical method mainly test a Boolean function of neighbor values against a predefined threshold. The digital method usually applies the Prewit filter to compute the gradients of node in x and y directions, if the synthetic gradient exceeds a threshold, the node is boundary node. The final method is based on the training pattern classifier, which attempts to sort all nodes into a bidirectional set of similar and dissimilar values. However, for optimal performance, all the above schemes require more information greater than coordinate, so we adopt a simple method only using coordinate in x and y direction, but its performance highly depends on the value of step-size parameter and people's logical decision.

2) ISOLINES DRAWING

Currently, isoline monitoring technology has two kinds: one is converging all sensor nodes and then generating isoline on the client-side, which suffering from heavy transmission; another is returning a portion of isoline nodes, then fitting the isoline on the client-side. Based on this reasoning, in [41], the authors propose a algorithm based on Delaunay triangle net, computing the coordinate of equivalent points using interpolating method if there are equivalent points in each edge of triangles. Then tracing and drawing equivalent points to create isolines. At last, smoothed isolines are formed based on cubic Bezier curve. Besides, [42] improves the drawback of not passing the given points of the traditional cubic B-spline smooth method. Control points are solved by the constructed equations through the boundary condition and the LU decomposition method. The smoothed isoline can be obtained by interpolation of cubic B-spline curves, which has high accuracy for passing all of the given points. But the computational overheads based on curve is expensive. So we use some simple wide-adopted interpolation methods to predict the values of some virtual inner nodes, such as cubic, nearest, invdist.

3) ESTIMATION FOR LEAKAGE SOURCE

Quickly detecting the accurate localization of gas leakage source are crucial for emergency responses, [43] proposes an effective algorithm to localize a continuous gas leakage source. They firstly obtain the possible source locations by reverse derivation from the diffusion model using Lambert W function, and then the most possible source location is determined based on prior information. Finally, the most possible source location is incorporated into the localization result, which is updated in a distributed sequential manner by adaptive weighted summation. Besides, [44] proposes a new algorithm assuming that the release of chemical gas can be modeled by a Gaussian-puff model, they use Taylor series approximation to linearize the measurement model. Then based on Gauss-Markov theorem they use the best linear unbiased estimator (BLUE) to estimate the location of leakage source. Except for [44], [45] also proposes a new algorithm to detect and localize multiple sources simultaneously based on Kalman filter and hypothesis testing. To decrease the computational complexity of the traditional algorithm, which rises exponentially with the number of source increases, they used greedy approach to iteratively detect potential sources one by one utilizing the measurement residuals of sensors. Besides, a gas leakage source localization (GLSL) algorithm based on distributed minimum mean squared error (D-MMSE) sequential estimation is proposed in [46]. However, these methods require some high mathematical skills. So we come up with a new idea derived from the interpolation technology adopted in drawing isolines, but sometimes the number of interpolation points is so large that greater calculation is required.

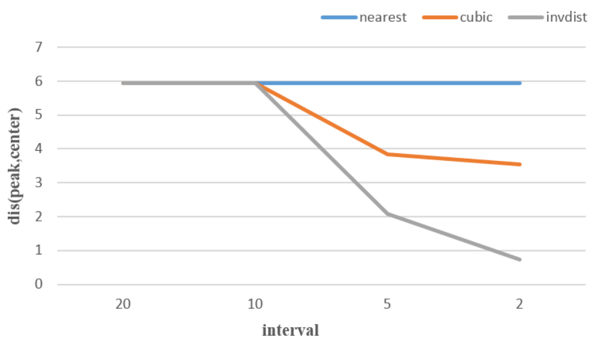


FIGURE 21. Distance between peak and center.

VII. CONCLUSION

The rapid development of WSNs has been of much recent interest, which results in a crucial problem about energy-efficient that people have to face. People do everything to save energy, we are no exception. In order to obtain abnormal field when poisonous gas occurs leakage, we comes up with an activating technology where sleeping sensor nodes do not need to detect environment and only wait to be activated by other abnormal nodes, which is energy-saving. When sensor node that is in the active state captures abnormal information, it transforms these messages along a shortest path through Dijkstra algorithm to the sink node. After receiving the current abnormal information of the network, we further propose to extract boundary nodes, draw isolines and estimate the location of leakage source in the sink. As to the third point, we adopt interpolation technology to estimate the leakage source for the first time. The result of our experiments shows that our method is practicable. In this paper, we can only achieve to track continuous objects in 2-D WSNs and in future, we will consider 3-D WSNs.

REFERENCES

[1] D. Alanis, P. Botsinis, S. X. Ng, and L. Hanzo, "Quantum-assisted routing optimization for self-organizing networks," *IEEE Access*, vol. 2, pp. 614–632, 2014.

[2] Y. Liu, J.-S. Fu, and Z. Zhang, "k-nearest neighbors tracking in wireless sensor networks with coverage holes," *Pers. Ubiquitous Comput.*, vol. 20, no. 3, pp. 431–446, 2016.

[3] S.-C. Tu, G.-Y. Chang, J.-P. Sheu, W. Li, and K.-Y. Hsieh, "Scalable continuous object detection and tracking in sensor networks," *J. Parallel Distrib. Comput.*, vol. 70, no. 3, pp. 212–224, 2010.

[4] J. Zhu and Y. Zou, "Cognitive network cooperation for green cellular networks," *IEEE Access*, vol. 4, pp. 849–857, 2016.

[5] B. Huang, W. Wu, and T. Zhang, "An improved connectivity-based boundary detection algorithm in wireless sensor networks," in *Proc. IEEE Conf. Local Comput. Netw.*, Oct. 2013, pp. 332–335.

[6] K. A. M. K. Azmi, W. A. T. Wan Abdullah, and Z. A. Ibrahim, "Hough transform method for track finding in center drift chamber," in *Proc. AIP Conf.*, 2016, p. 1704.

[7] B. Zhou and Y. He, "Fast circle detection using spatial decomposition of Hough transform," *Int. J. Pattern Recognit. Artif. Intell.*, 2016, doi: http://dx.doi.org/10.1142/S0218001417550060.

[8] W. Cho and J. Jeon, "Edge detection in wavelet transform domain," in *Proc. Workshop Frontiers Comput. Vis.*, 2015, pp. 1–4.

[9] H. Yang, X. Li, Z. Wang, W. Yu, and B. Huang, "A novel sensor deployment method based on image processing and wavelet transform to optimize the surface coverage in WSNs," *Chin. J. Electron.*, vol. 25, no. 3, pp. 495–502, 2016.

[10] H. Du, Z. Huang, and J. He, "The research of image edge detection based on wavelet-transform," *Autom. Instrum.*, vol. 25, no. 1, pp. 13–14, 2016.

[11] X. Wu, H. Chen, Y. Wang, L. Shu, and G. Liu, "BP neural network based continuous objects distribution detection in WSNs," *Wireless Netw.*, vol. 22, no. 6, pp. 1917–1929, 2016.

[12] F. U. Guangjie, D. Zhao, Q. Zhao, and S. Wang, "Research on detection and location system of pipeline leakage based on acoustic wave technology," *Mod. Electron. Techn.*, vol. 38, no. 17, pp. 101–102, 2015.

[13] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. 33rd Annu. Hawaii Int. Conf. Syst. Sci.*, 2000, pp. 1–10.

[14] Z. Zhou, D. Zhao, L. Shu, and H. C. Chao, "Efficient multi-attribute query processing in heterogeneous wireless sensor networks," *J. Internet Technol.*, vol. 15, no. 5, pp. 699–712, 2014.

[15] K. Yuan, Q. Ling, and Z. Tian, "Communication-efficient decentralized event monitoring in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 8, pp. 2198–2207, Aug. 2015.

[16] H. Park, S. Oh, E. Lee, S. Park, S. H. Kim, and W. Lee, "Selective wakeup discipline for continuous object tracking in grid-based wireless sensor networks," in *Proc. Wireless Commun. Netw. Conf. (WCNC)*, 2012, pp. 2179–2184.

[17] B. Huang, W. Wu, G. Gao, and T. Zhang, "Recognizing boundaries in wireless sensor networks based on local connectivity information," *Int. J. Distrib. Sensor Netw.*, vol. 2014, no. 3, pp. 1–12, 2014.

[18] D. K. Fan and P. Shi, "Improvement of Dijkstra's algorithm and its application in route planning," in *Proc. Int. Conf. Fuzzy Syst. Knowl. Discovery*, 2010, pp. 1901–1904.

[19] *Dijkstra's Algorithm*, accessed on Sep. 3, 2016. [Online]. Available: https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

[20] M. Abo-Zahhad, R. R. Gharieb, S. M. Ahmed, and A. B. El-Baset Donkol, "Edge detection with a preprocessing approach," *J. Signal Inf. Process.*, vol. 5, no. 4, pp. 123–134, 2014.

[21] M. Sofka, K. Ralovich, N. Birkbeck, and J. Zhang, "Integrated detection network (IDN) for pose and boundary estimation in medical images," *Proc. IEEE*, vol. 48, no. 1, pp. 294–299, Jan. 2011.

[22] Chenchunsy. *The Boundary Extraction of Point Cloud*, accessed on Jul. 7, 2016. [Online]. Available: http://zhidao.baidu.com/question/205978618.html

[23] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The Quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, pp. 469–483, 1996.

[24] X. Meng, J. Li, Q. Yang, Q. Cai, and Q. Chen, "Complex conforming delaunay triangulation," *Sci. China Inf. Sci.*, vol. 53, no. 6, pp. 1130–1140, 2010.

[25] D. M. Yan, W. Wang, B. Lévy, and Y. Liu, "Efficient computation of clipped Voronoi diagram for mesh generation," *Comput.-Aided Design*, vol. 45, no. 4, pp. 843–852, 2013.

[26] *Inverse Distance Weighting*, accessed on Dec. 19, 2015. [Online]. Available: https://en.wikipedia.org/wiki/Inverse_distance_weighting

[27] J. G. Wolff, "Autonomous robots and the SP theory of intelligence," *IEEE Access*, vol. 2, no. 1, pp. 1629–1651, Jan. 2014.

[28] H. Zhou, S. Xia, M. Jin, and H. Wu, "Localized and precise boundary detection in 3-D wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 23, no. 6, pp. 1742–1754, Dec. 2015.

[29] C. Farah, C. Zhong, M. Worboys, and S. Nittel, "Detecting topological change using a wireless sensor network," in *Proc. Int. Conf. Geograph. Inf. Sci.*, 2008, pp. 55–69.

[30] S. Escolar, S. Chessa, J. Carretero, and M.-C. Marinescu, "Cross layer adaptation of check intervals in low power listening mac protocols for lifetime improvement in wireless sensor networks," *Sensors*, vol. 12, no. 8, pp. 10511–10535, 2012.

[31] W.-C. Chu and K.-F. Ssu, "Location-free boundary detection in mobile wireless sensor networks with a distributed approach," *Comput. Netw.*, vol. 70, no. 10, pp. 96–112, 2014.

[32] V. Vaidehi, U. S. Selvan, J. Jayendran, and K. Praveen, "Redundant node deactivation by scheduling in wireless sensor networks," in *Proc. Recent Trends Inf. Technol. (ICRTIT)*, 2011, pp. 613–617.

[33] H. F. Lin, A. Jiang, D. Bai, and Y. F. Liu, "Energy-efficient node scheduling for dense wireless sensor networks using Voronoi diagram," *Adv. Technol. Manuf. Syst. Ind.*, vol. 236, pp. 1054–1066, Nov. 2012.

[34] L. J. Sun, J. Wei, J. Guo, F. Xiao, and R. C. Wang, "Node scheduling algorithm for heterogeneous wireless sensor networks," *Tien Tzu Hsueh Pao/Acta Electron. Sin.*, vol. 42, no. 10, pp. 1907–1912, 2014.

[35] Q. Zhang, L. Sun, J. Guo, F. Xiao, and Wang, "A grid-based coverage-preserving node scheduling algorithm," *J. Comput. Res. Develop.*, vol. 48, no. 7, pp. 111–115, 2011.

[36] Y. M. Miao, "Cluster-head election algorithm for wireless sensor networks based on leach protocol," *Appl. Mech. Mater.*, vols. 738–739, pp. 19–22, Mar. 2015.

[37] J. Liu, P. Cheung, F. Zhao, and L. Guibas, "A dual-space approach to tracking and sensor management in wireless sensor networks," in *Proc. 1st ACM Int. Workshop Wireless Sensor Netw. Appl.*, 2002, pp. 131–139.

[38] D. Ganesan, D. Estrin, and J. Heidemann, "Dimensions: Why do we need a new data handling architecture for sensor networks?" in *Proc. ACM Workshop Hot Topics Netw.*, 2002, pp. 143–148.

[39] R. Nowak and U. Mitra, "Information processing in sensor networks," in *Boundary Estimation Sensor Networks: Theory Methods*. Berlin, Germany: Springer, 2003, pp. 80–95.

[40] K. K. Chintalapudi and R. Govindan, "Localized edge detection in sensor fields," *Ad Hoc Netw.*, vol. 1, nos. 2–3, pp. 273–291, 2003.

[41] Y. Jiang, D. U. Bin, L. U. Jun, and P. Wang, "Algorithm of drawing isoline based on Delaunay triangle net," *Appl. Res. Comput.*, vol. 27, no. 1, pp. 101–103, 2010.

[42] T. Xu, Z. Cao, and G. Yang, "An isoline smooth algorithm and its application based on cubic b-spline interpolation," *Int. J. Adv. Comput. Technol.*, vol. 4, no. 14, pp. 43–50, 2012.

[43] M. Cao, Q. Meng, Y. Wu, and M. Zeng, "Distributed sequential adaptive weighted localization of a gas-leakage source using a wireless sensor network," in *Proc. Control Decision Conf.*, 2013, pp. 3686–3691.

[44] X. X. Qin, W. J. Wang, and C. C. Yin, "A localization algorithm for gas leakage source based on wireless sensor network," *Software*, vol. 34, no. 1, pp. 111–115, 2013.

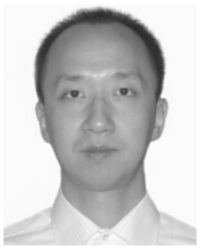
[45] Y.-W. Bi and Y.-T. Gu, "Study on gas leak source detection and localization based on greedy approach," *Comput. Simul.*, vol. 1, no. 1, pp. 286–289, 2014.

[46] Z. Yong, Q. Meng, W. U. Yuxiu, and Z. Ming, "Gas leakage source localization algorithm based on distributed MMSE sequential estimation," *Chin. J. Sens. Actuators*, vol. 27, no. 1, pp. 128–134, 2014.

FEI LEI is currently an Associate Professor with the Faculty of Information Technology, Beijing University of Technology, China. His research interests include wireless sensor networks and digital image processing.

DENG ZHAO is currently pursuing the master's degree with the School of Information Engineering, China University of Geosciences, Beijing. Her research interests include wireless sensor networks and spatial and temporal database.

LEI YAO is currently pursuing the master's degree with the Faculty of Information Technology, Beijing University of Technology, China. Her research interests include wireless sensor networks.



YUCONG DUAN is currently a Full Professor with the College of Information Science and Technology, Hainan University, China. His research interests include theoretical and empirical software engineering, model driven software development, knowledge engineering, ontology modeling, data cleaning of very large database, cognitive linguistics, service computing, SOA, and cloud computing.