This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2007 proceedings.

OFRD:Obstacle-Free Robot Deployment Algorithms for Wireless Sensor Networks

Chih-Yung Chang[+], Hsu-Ruey Chang[+], Chen-Chi Hsieh[+], Chao-Tsun Chang[*]
[+]Tamkang University, Taiwan
[*]Hsiuping Institute of Technology, Taiwan
[+]cychang@mail.tku.edu.tw, [+]{hrchang, jenchi}@wireless.cs.tku.edu.tw, [*]cctas@mail.hit.edu.tw

***Abstract*—Node deployment is an important issue in Wireless sensor networks (WSNs). Sensor nodes should be efficiently deployed in a predetermined region in a low cost and high coverage quality manner. Random deployment is the simplest way for deploying sensor nodes but may cause the unbalanced deployment and therefore increase the hardware cost. This paper presents an efficient obstacle-free robot deployment algorithm, called OFRD which involves the design of node placement policy, snake-like movement policy, and obstacle handling rules. By applying the proposed OFRD, the robot rapidly deploys near-minimal number of sensor nodes to achieve full sensing coverage even though there exist unpredicted obstacles. Performance results reveal that OFRD outperforms the existing robot deployment mechanism in terms of power conservation and obstacle resistance, and, therefore achieves a better deployment performance.**

Keywords-deployment; repair; patrol; sensor network; robot

I. INTRODUCTION

Wireless Sensor Networks compose of many sensor nodes embedded with simple process, fewer memory, tiny sensing material, and energy-limited battery. In literature, existing deployment algorithms can be classified into three categories: stationary sensor [1], mobile sensor [2], and mobile robot [3, 4]. Several random deployment schemes [6] were proposed for the deployment of stationary sensors. The random deployment is simple and easy to be implemented. However, to ensure full coverage, the number of deployed sensors is extremely larger than the one that is actually required. Deploying stationary sensors randomly may result in an inefficient WSN where some areas is densely deployed but the other areas may be deployed with a low density. The dense deployment in some areas increases the hardware cost whereas the sparse deployment in the other areas results in coverage holes or network partition.

Some other study [2] developed mechanism to cope with the coverage problem in a mobile WSN. Mobile sensors cooperatively compute their target locations according to the information about holes after an initial phase of random deployment of stationary sensors and then move to target locations to heal the existed coverage holes. However, hardware cost can not be saved for those areas that have been densely deployed with stationary sensors.

Another deployment alternative [3, 4] uses the robot to deploy static sensors in a given region. The robot explores the environment and deploys a stationary sensor on the target location from time to time. In [4], the robot deploys the sensors according to the predefined direction priority of south, west, north, and east. Each sensor counts the time interval that the robot does not visit for each direction. Deployed sensors within the communication range of the robot may guide the robot's movement by suggesting a suitable direction with maximal time interval to the robot. As the robot received some suggestions, it integrates the suggestion and selects the best direction for patrol or sensor deployment. However, the approach can not guarantee full coverage and may cause too much sensing redundancy if the robot encounters obstacles.

This paper presents an efficient robot deployment algorithm OFRD which involves the design of node placement policy, snake-like movement policy, and obstacle handling rules. Simulation results reveal that the proposed OFRD deploys fewer static sensors but achieve higher coverage percentage than the existing deployment algorithm.

II. NETWORK ENVIRONMENT AND BASIC CONCEPTS

2.1 Network Environment

This paper considers a single robot that carries limited static sensor nodes and embedded with a compass through which the robot is aware of its moving direction. Initially, the robot is assumed to be located at the left corner of the monitoring region. Let r_c and r_s denote the communication and sensing ranges, respectively. Herein, we assume r_c is larger than $\sqrt{3}r_s$.

2.2 Basic Concepts

An optimal robot deployment refers to the deployment that the robot deploys minimal number of sensors but achieve full-coverage purpose. To achieve the optimal deployment, the overlapped sensing region of neighboring sensor nodes should be strictly controlled. Figure 1(a) illustrates the basic requirement for optimal deployment. Let nodes A, B, and C be the three neighboring sensors. The optimal deployment can be reached if the three sensor nodes intersect with each other at one point. In this situation, the distance of any pair of A, B, and C exactly equals to $\sqrt{3}r_s$. Based on this, *the deployment policy* arranges the robot deploys a sensor every $\sqrt{3}r_s$.

In addition to *the deployment policy*, a snake-like movement policy is employed for robot's movement. Figure 1(b) depicts the snake-like movement where the robot deploys a sensor every $\sqrt{3}r_s$ distance.

Furthermore, the proposed OFRD aims to efficiently overcome the unpredicted obstacle and develops obstacle handling rules to alleviate the impact of obstacles on the execution of deployment task. As shown in Fig. 2, the black

blocks represent three obstacles with different shapes and the blue lines denote the trajectory of the robot's moves by applying the proposed OFRD. To highlight the movement trajectory, sensors deployed in the WSN are not shown in the Figure. The trajectory shows that OFRD takes into consideration the unpredicted obstacles and achieves full coverage even though the monitoring region exists obstacles.

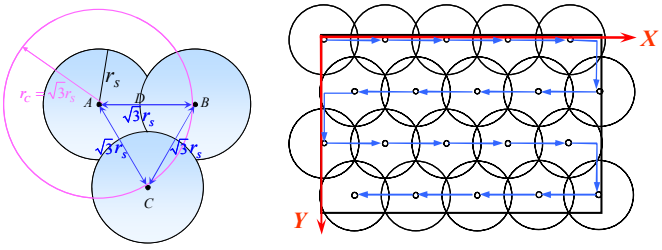


Fig. 1. (a) The optimal deployment of the three nearby sensors *A*, *B*, and *C*. (b) The snake-like movement deployment.

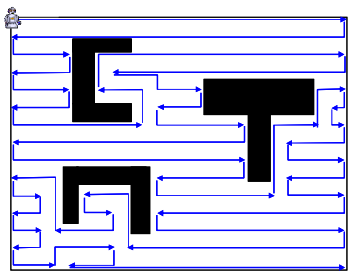
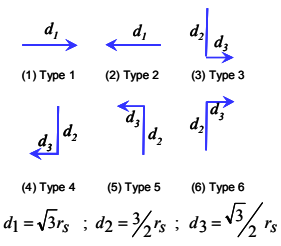
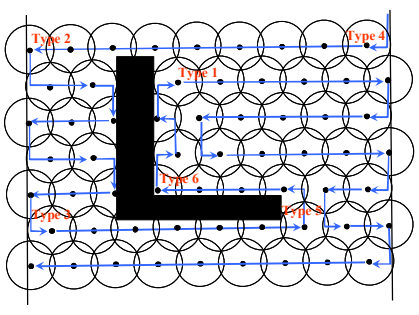


Fig. 2. The deployment by applying the obstacle handling mechanism.



(a) Six legal patterns for basic movement.



(b) A scenario of applying six types of basic movement to overcome the obstacle.

Fig. 3. Six types of legal patterns for basic movements and their usages.

III. Obstacle-Free Deployment Mechanism

3.1 Legal Movement Patterns

The robot will deploy a sensor node after each movement of distance $\sqrt{3}r_s$. To achieve the optimal deployment, the movement of robot should be one of the six legal patterns as shown in Fig. 3(a). The six types of basic movement are referred to the *legal patterns for basic movement*. Types 1 and 2 are used when the robot moves toward east and west directions, respectively. As shown in Fig. 1(a), the sensor nodes deployed on the *i*th row are located on the perpendicular bisector of two neighboring sensors deployed on the (*i*-1)th row. As the robot encounter boundary or obstacle, it should deploy the sensor on next row. That is, the robot should move toward the south. Type 3 will be used when the robot moves toward west direction but encounters left boundary or obstacle. In this case, the robot will move toward south for a distance of $(3/2)r_s$, then moves toward east for a distance of $(\sqrt{3}/2)r_s$. Similarly, type 4 is used when the robot moves toward east but encounters right boundary or obstacle. To overcome the unpredicted obstacles, sometimes the robot would moves toward north direction. Types 5 and 6 are used when the robot tries to overcome obstacle and moves toward north direction. Figure 3(b) shows the scenarios of applying six types of basic movements.

3.2 Simple Snake-Like Robot Deployment

In the snake-like movement, the robot stays in one of the two possible states: East and West. The East and West states denote that the robot is currently moving toward east and west directions, respectively. Each state has two movement direction options with different priorities to guide the robot to the promising direction of the next movement as shown in Table I. In both East and West states, the Prefer Direction 1 has a higher priority which enables the robot moves along east and west directions, respectively. The Prefer Direction 2 has a lower priority and will be applied in case that the movement in Prefer Direction 1 is failure. The Prefer Direction 2 will guide the robot moving toward south. As soon as the Prefer Direction 2 has been applied, the robot's state should be changed.

Table I：Simple Snake-like Deployment.

States	Prefer Direction 1	Prefer Direction 2
East	→(Type 1)	↲(Type 4)
West	←(Type 2)	↳(Type 3)

For simplicity, the robot is initially located at the left-up corner of the monitoring region. The initial state of robot will be East state. According to Table I, the robot will determine the direction of the next movement according to Prefer Direction 1 and hence it moves toward east for a distance $\sqrt{3}r_s$. The robot then deploys a sensor and repeatedly moves toward east direction and deploys a sensor until it encounters the right boundary. Since the right boundary may cause the east movement failure, the robot then makes decision according to Prefer Direction 2 and hence moves toward south direction for a distance $(3/2)r_s$ and then moves toward west direction for a distance $(\sqrt{3}/2)r_s$. As soon as the robot changes its movement direction, the state of robot also changes from East to West. After that, the robot makes decision according to the Prefer direction 1 and hence moves toward west direction until the left boundary is encountered. Note that the robot deploys a sensor node every $\sqrt{3}r_s$ distance. the East and West states can

be taken place in turns in the snake-like movement as shown in Fig. 1(b).

3.3 Obstacle-Free Snake-Like Robot Deployment

A. The Impact of Obstacles

This subsection considers additionally the existence of obstacles and develops an obstacle-free snake-like deployment mechanism. Figure 4 depicts two main challenges of the developed simple snake-like movement scheme. As shown in Fig. 4, the black region represents an obstacle and the directional blue line represents the trajectory of the robot's movement by applying the simple snake-like deployment mechanism proposed in previous subsection. As a result, the deployment remains two sensing holes *A* and *B* as shown in Fig. 4.

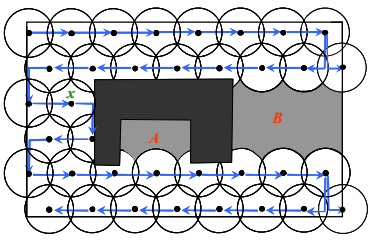


Fig. 4. Applying simple snake-like deployment results holes due to the existence of the obstacle.

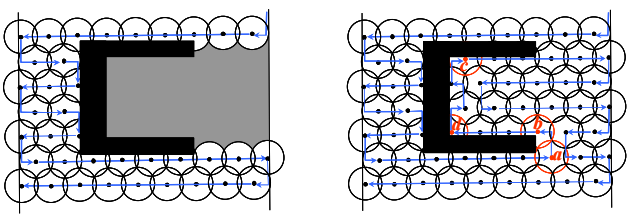


Fig. 5. (a) Applying the simple snake-like deployment scheme results a inner-concave sensing hole. (b) The proposed obstacle-free snake-like deployment mechanism can overcome the obstacle and achieves full coverage.

B. Obstacle Handling Rules

Assume that the robot stays in East state. The robot repeatedly applies type 1 movement and hence moves in east direction. As the robot encounters the right boundary, it checks Prefer Direction 2 according to the simple snake-like movement mechanism and applies type 4 movement. Therefore, the robot moves in south direction for a distance $(3/2)r_s$ firstly and then moves in west direction for a distance $(\sqrt{3}/2)r_s$. However, moving in this way will results that there is no opportunity for the robot to visit the north and west directions to redeploy sensors in the existed hole. In order to overcome the obstacles, the robot should check whether or not there exists any sensing hole in the north or west directions. Therefore, the movement in north and west directions should be prior to the current Prefer direction 1.

Table II lists the *check directions* for the robot to further check if the *check direction* exists any sensing hole. In prior moving in Prefer Direction 1, the robot will check the *check direction* first and tries to move in the check direction. In case that there exist sensors deployed in the checked direction, the robot will then apply simple snake-like movement scheme and utilize Table I to decide the next movement direction.

There are two check directions for each state. In case that the robot stays in East state, it firstly checks the *Check direction 1*. In case that there is no sensor deployed in the West direction, the next movement direction will be West. Otherwise, the robot will check the *Check Direction 2*. If the north direction did not deploy any sensor, the robot will move in north direction in the next movement. If, fortunately, there does not exist hole in the two check directions, the robot will further apply simple snake-like movement scheme which utilizes Table I to determine the next movement direction. Situation that the robot stays in West state is similar to that in East state and is omitted herein.

Figure 5(a) gives an example that contains an obstacle in the monitoring region. Applying simple snake-like deployment algorithm will results in a hole region (marked by the gray color) containing no sensor. Figure 5(b) exhibits the concept of obstacle-free snake-like deployment. As the robot stays in the East state and visits location *a*, it checks Check Direction 1(West). Since there is a deployed sensor in the West direction, the movement in Check Direction 1 is failure. The robot further checks Check Direction 2(North) and finds that there is a hole in the north direction. Then the robot moves in north direction and uses type 5 movement which moves in north direction first and then west direction. As the robot arrives location *b*, it checks Check Direction 1 and moves in West direction. Following the rules described above, the robot will continue to move toward the West direction until it arrives location *d* and again the robot encounters the obstacle. Since the movement in Check Direction 1(West) is failure, the robot then checks Check Direction 2 and moves in north direction accordingly. During the movements from locations *d* to *c*, the movement in Check Direction 1 will be failure and therefore the robot moves according to Check Direction 2. By checking Check Direction 2 as shown in Table II, the robot uses types 5 and 6 movement patterns alternatively and moves from locations *d* to *c*. The movements after location *c* will base on simple snake like movement mechanism since both Check Direction 1 and Check Direction 2 are failure. Finally, the robot overcomes the impact of obstacle and achieves the goal of full coverage deployment.

Table II： Check Directions for overcoming the obstacle.

States	Check Direction 1	Check Direction 2	
East	←(Type 2)	↰(Type 5)	↱(Type 6)
West	→(Type 1)	↱(Type 6)	↰(Type 5)

Table III： Obstacle-Free Snake-Like Movement Rule.

States	Check Direction 1	Check Direction 2		Prefer Direction 1	Prefer Direction 2	
East	←	↰	↱	→	↳	↲
West	→	↱	↰	←	↲	↳

Table III summarizes the check directions and prefer directions in a priority order. As shown in Table III, the robot should select one of the six types movement as the movement pattern according to the priority of each movement type listed

in Table III. In general, if the robot stays in the East state, the six movement types have the following priorities:

Type 2 > Type 5 > Type 6 > Type 1 > Type 3 > Type 4

(←) (↰) (↱) (→) (↳) (↲)

On the other hand, if the robot stays in the West state, the six movement types have the following priorities:

Type 1 > Type 6 > Type 5 > Type 2 > Type 4 > Type 3

(→) (↱) (↰) (←) (↲) (↳)

Note that if the robot fails to move in a certain direction with higher priority, the robot will try the next higher priority until there is a successful movement. The following four rules summarize the abovementioned obstacle handling algorithm. A try of movement direction is said to be ***failure*** if there exist a deployed sensor, obstacle, or boundary of monitoring region in that direction.

The robot moves and deploys sensor nodes according to the following four rules in order. Rules 1 and 2 are mainly designed for handling obstacles whereas Rules 3 and 4 are designed for snake-like movement in the environment without obstacle. The robot checks the rules from rule 1 to rule 4 in order and executes one of the four rules to select a direction for the next movement. The robot moves towards to the selected direction for a predefined distance as shown Fig. 3(a) and then deploys a static sensor. Then the robot determines the next moving direction by checking the following four rules again.

/* Rules 1 and 2 are designed for overcoming obstacle */

Rule1: The robot checks Check Direction 1 (which is opposite direction to Prefer Direction 1) for possible movement. If the try in this direction is failure, the robot executes the next rule. Otherwise, the robot will move toward the Check Direction 1 for $\sqrt{3}r_s$ distance.

Rule2: The robot tries to move in Check Direction 2. If the try in this direction is failure, the robot checks Rule 3 subsequently. Otherwise, the robot moves toward the Check Direction 2 for $(3/2)r_s$ distance.

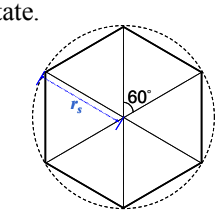
/* Rules 3 and 4 are designed for snake-like movement */

Rule3: The robot checks Prefer Direction 1 for possible movement. If the try in this direction is failure, the robot executes Rule 4. Otherwise, the robot moves toward the Prefer Direction 1 for $\sqrt{3}r_s$ distance.

Rule4: The robot checks Prefer Direction 2 for possible movement. If the try in this direction is failure and the deployment is not terminated, the robot will go back to the location of previous deployed sensor and checks the four rules again in order. Otherwise, the robot moves toward the Check Direction 2 for $(3/2)r_s$ distance.

A region without any deployed sensor can be treated as a coverage hole without sensing capability. Assume the robot stays in East state. Applying the simple snake-like movement (Rules 3 and 4), the robot will moves from West to East until it encounters the right boundary and then from North to South direction. Therefore, the most difficult for robot deployment is that there exists obstacle so that a hole appeared in the West or North directions. However, if there is a hole in West or North directions, the robot may apply Rules 1 and 2, examines the Check Directions including West and North directions, and then moves back toward the West and North directions to deploy the sensors in the hole region. Therefore, the four rules presented herein can overcome the existence of unpredicted obstacle and achieve the purpose of full coverage deployment.

As the obstacle free snake-like deployment algorithm involves the consideration of two Check Directions, the constraint that the robot should initially starts its movement at left up corner could be released. Even though the robot initially starts its movement at the central location of the monitoring region, the robot may apply Rules 1 and 2 to move toward West and North directions to deploy the sensors and then moves East and South to achieve the purpose of full coverage deployment. The state of robot depends on the initial location. If the robot starts its movement in east (or west) boundary of the monitoring region, the robot will stay in East (or West) state.



$$Area = \Delta \times 6 = \frac{\sqrt{3}}{4} r_s^2 \times 6 = \frac{3\sqrt{3}}{2} r_s^2$$

Fig. 6. In analysis, a hexagon region is used to represent the sensing area of one sensor.

IV. Analysis of Deployment Efficiency

4.1 Without the Obstacle Environment

This subsection analyzes the performance efficiency of OFRD algorithm in a non-obstacle environment in terms of the number of deployed sensors. As shown in Fig. 6, the dotted circle denotes the sensing range r_s. For simplicity of analysis, the maximal hexagon cell covered by the sensor is used to represent the sensing range of the sensor. The optimal deployment of an area is the one that is deployed with the minimal number of sensors but achieves full coverage purpose. The number of sensors deployed in an optimal deployment equals to the number of hexagon partitions in the area.

The area of each hexagon can be derived by expression (1).

$$Area = \Delta \times 6 = \frac{\sqrt{3}}{4} r_s^2 \times 6 = \frac{3\sqrt{3}}{2} r_s^2 \quad (1)$$

Let L and W denote the length and width of the monitoring region, respectively. The ideal number of sensor nodes deployed in the monitoring region can be derived by (2).

$$N_{ideal}(W,L) = \left\lceil \frac{W \times L}{Area} \right\rceil = \left\lceil \frac{W \times L}{3\sqrt{3} r_s^2 / 2} \right\rceil = \left\lceil \frac{2\sqrt{3}}{9 r_s^2} \times WL \right\rceil \quad (2)$$

In the real scenario, the robot may require to deploy one more sensor as it encounters the boundary of the monitoring region. Therefore, the robot requires to deploy more sensors than that in the ideal case. Expression (3) evaluates the number of sensors deployed by applying the OFRD mechanism in the worst case.

$$N_{Worst}^{OFRD}(W,L) = \left\lceil \frac{W \times L}{Area} \right\rceil + 2 \times \left\lceil \left(\frac{W}{2} + L \right) \right\rceil = \left\lceil \frac{2\sqrt{3}}{9 r^2} \times W \times L \right\rceil + 2 \times \left\lceil \left(\frac{W}{2} + L \right) \right\rceil \quad (3)$$

In considering the average case, rather than the worst case, the number of sensors deployed nearby the boundary highly depends on the distance between the latest deployed sensor and the boundary.

Let $H = \left\lceil \frac{L}{height\_of\_a\_hexagon\_cell} \right\rceil$ and $P = \left\lceil \frac{W}{height\_of\_a\_hexagon\_cell} \right\rceil$.

Since the probabilities that each row and each column require to deploy one more sensor are 1/2 and 1/3, respectively, expression (4) reflects the average number of deployed sensors by applying OFRD.

$$N_{approxiate}^{OFRD}(W,L) = \left\lceil \frac{W \times L}{Area} \right\rceil + 2 \times \left(\left\lceil \frac{1}{2} \times H \right\rceil + \left\lceil \frac{1}{3} \times \mathrm{P} \right\rceil \right) = \left\lceil \frac{2\sqrt{3}}{9r_s^2} \times W \times L \right\rceil + \lceil H \rceil + \left\lceil \frac{2}{3} \times \mathrm{P} \right\rceil \quad (4)$$

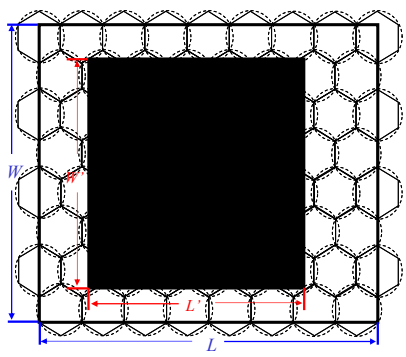


Fig. 7. The monitoring region contains a rectangle obstacle with size *W'*L'*.

4.2 With the Obstacle Environment

As shown in Fig. 7, let *L* and *W* respectively denote the length and width of the monitoring region and *L'* and *W'* respectively denote the length and width of a given obstacle. In calculating the number of required sensors in the ideal case, expression (2) is utilized to measure the minimal number of deployed sensors in the non-obstacle WSN. Expression (5) calculates the ideal number of deployed sensor in the monitoring region containing an *L'*W'* sized obstacle.

$$N_{ideal} = N_{ideal}(W,L) - N_{ideal}(W',L') = \left\lceil \frac{W \times L - W' \times L'}{Area} \right\rceil = \left\lceil \frac{W \times L - W' \times L'}{3\sqrt{3}r^2/2} \right\rceil = \left\lceil \frac{2\sqrt{3}}{9r^2} \times (W \times L - W' \times L') \right\rceil \quad (5)$$

The number of deployed sensors highly depends on the boundary locations of the obstacle. In the worst case, the robot should deploy one more sensor node at it encounters boundaries every two rows or every three columns. Consequently, expression (6) measures the number of deployed sensors by applying the OFRD in an obstacle environment.

$$N_{Worst}^{OFRD}(W',L') = \left\lceil \frac{W \times L}{Area} \right\rceil - \left\lfloor \frac{W' \times L'}{Area} \right\rfloor + 2 \times (H + h' + \mathrm{P} + \mathrm{p}') = \left\lceil \frac{2\sqrt{3}}{9r_s^2} \times W \times L \right\rceil - \left\lfloor \frac{2\sqrt{3}}{9r_s^2} \times W' \times L' \right\rfloor + 2 \times (H + h' + \mathrm{P} + \mathrm{p}') \quad (6)$$

, where $H' = \left\lceil \frac{L'}{height\_of\_a\_hexagon\_cell} \right\rceil$ and $P' = \left\lceil \frac{W'}{height\_of\_a\_hexagon\_cell} \right\rceil$.

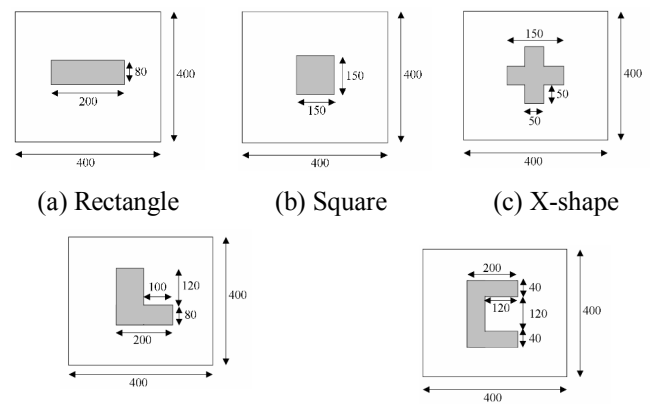
Similar to expression (4), expression (7) evaluates the approximate number of the deployed sensors by applying the proposed OFRD algorithm.

$$N_{approximae}^{OFRD}(W',L') = \left\lceil \frac{W \times L}{Area} \right\rceil - \left\lfloor \frac{W' \times L'}{Area} \right\rfloor + 2 \times \left(\left\lceil \frac{1}{2} \times (H + h) \right\rceil + \left\lceil \frac{1}{3} \times (\mathrm{P} + \mathrm{p}') \right\rceil \right) = \left\lceil \frac{2\sqrt{3}}{9r_s^2} \times W \times L \right\rceil - \left\lfloor \frac{2\sqrt{3}}{9r_s^2} \times W' \times L' \right\rfloor + H + h + \left\lceil \frac{2}{3} \times (\mathrm{P} + \mathrm{p}') \right\rceil \quad (7)$$

V. PERFORMANCE STUDY

The simulation environment is set up similar to [5]. The network size is 400*400m^2. The initial location of the robot is located at the left-top corner of the monitoring region. The communication and sensing ranges are set at 40m and 20m, respectively. The energy consumptions for packet transmission, packet reception, and idle listening are set at 0.075J/s, 0.030J/s, and 0.025J/s, respectively. The initial energy of each sensor is set by 32400J. The total energy and the speed of the robot are 64800J and 3m/s, respectively. The mobility cost is set at 8.267J/m which refers to previous work [5]. Each simulation result is obtained from the average of 10 independent runs.

The proposed OFRD is compared with CED[4] in terms of the number of deployed sensor and the coverage percentage in a WSN environment with or without obstacle. Five different shapes of obstacles are considered in the simulation as shown in Fig. 8.



(a) Rectangle (b) Square (c) X-shape

(d) L-shape (e) C-shape

Fig. 8. Obstacles with various shapes are considered in the simulation environment.

Figure 9 depicts the deployment by applying the proposed OFRD in an environment containing a C-shape obstacle. It is observed that the robot can efficiently overcome the obstacle and deploys 163 sensors which is the minimal number of sensors to achieve full coverage purpose.

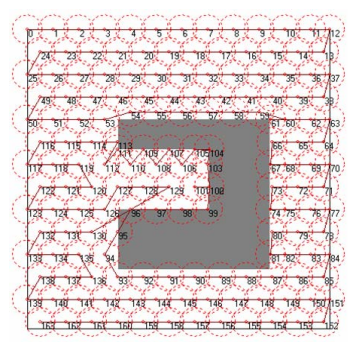


Fig. 9. Screenshots of executing the OFRD mechanism.

Table IV : Comparison of the number of sensors deployed by applying OFRD and CED schemes in different environments.

	No obstacle	Rectangle obstacle	Square obstacle	X-shape obstacle	L-shape obstacle	C-shape obstacle
OFRD	174	166	162	171	160	163
CED	400	321	404	388	364	370

Table IV investigates the number of sensors required by applying the OFRD and CED mechanisms. Environments that contain various shapes of obstacles are considered. According to the predefined direction priority of south, west, north, and east, the CED deploys a sensor when the robot moves out of the sensing region of the deployed sensor. Since the CED deploys sensor by negotiating with the closest deployed sensor, the newly deployed sensor may have significant redundant sensing range with the neighboring sensors other than the closest one. In comparison, the proposed OFRD significantly reduces the number of deployed sensors and achieves full coverage purpose.

Different shapes of the obstacle may impact the number of deployed sensors even though the same deployment scheme is employed. Figure 10 investigates the performance of OFRD and CED under the environment containing different shapes of obstacles. The *NS-OFRD* and *NS-CED* denote the number of deployed sensors by applying OFRD and CED mechanisms. In comparison, the proposed OFRD significantly reduces the number of deployed sensors and therefore outperforms CED in all cases. In particular, OFRD maintains similar numbers of deployed sensors in all cases of obstacle shapes. On the other hand, Fig. 15 also investigates the coverage percentage of the monitoring region. The *CP-OFRD* and *CP-CED* denote the coverage percentages by applying OFRD and CED, respectively. In general, the OFRD keeps the coverage percentage above 95% and outperforms CED in the cases of X-shape and C-shape obstacles.

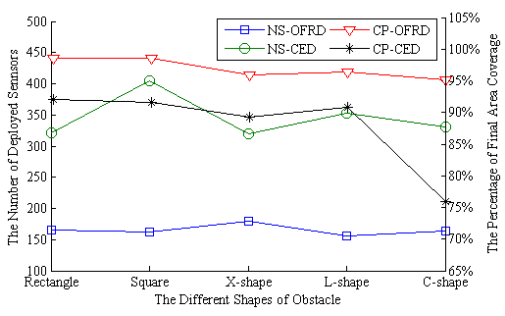


Fig. 10. The *OFRD* reduces the number of deployed sensors and maintains near-full coverage in different environments.

The OFRD classifies the basic movement into six patterns. Figure 11 depicts the usages of the six movement patterns. In the experiments, movement types 1 and 2 are used most frequently. The types 4, 5, and 6 are frequently used when the shapes of obstacles are irregular. For example, these types of movements are frequently adopted by the robot when the obstacles are X-shape, L-shape, and C-shape.

Figure 12 measures the usage of the proposed four rules designed in the OFRD. Different shapes of obstacles are considered in the environment. Recall that Rules 1 and 2 are designed for overcoming obstacle and Rules 3 and 4 are designed for snake-like movement. In the environment without obstacle, the OFRD applies Rules 3 and 4 to achieve full coverage. As soon as there exist different shapes of obstacles in the environment, Rules 1 and 2 are applied.

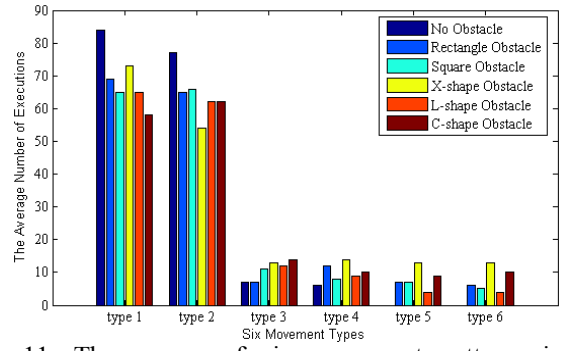


Fig. 11. The usages of six movement patterns in the environment containing different shapes of obstacles.

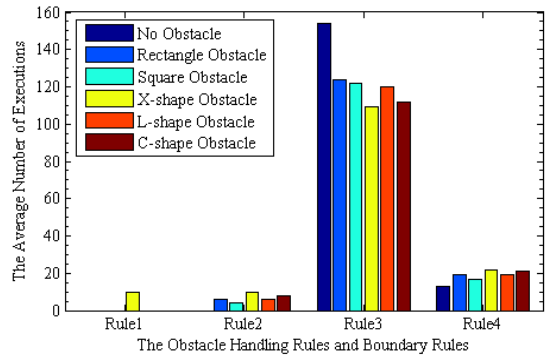


Fig. 12. The usages of the four Rules designed in the OFRD in the environment containing different shapes of obstacles.

VI. CONCLUSIONS

This paper proposes an obstacle-free robot deployment mechanism that deploys the monitoring region with near-minimal number of sensors and likely achieves the full coverage purpose. The proposed OFRD involves the deployment policy, snake-like movement policy, and the obstacle handling rules that help the robot resist the unpredicted obstacles. Performance results reveal that the proposed OFRD outperforms the existing CED mechanism in terms of the number of deployed sensor and the coverage percentage.

REFERENCES

[1] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan and K. K. Saluja, "Sensor Deployment Strategy for Target Detection," *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications* (WSNA'02), pp. 42-48, Atlanta, Georgia ,USA September 2002.

[2] T. L. Wong, T. Tsuchiya, and T. Kikuno, "A Self-Organizing Technique for Sensor Placement in Wireless Micro-Sensor Networks," *Proceedings of the 18th IEEE Internation Conference on Advanced Information Networking and Applications*, (AINA'04), vol. 1, pp. 78-83, Fukuoka, Japan, March 2004.

[3] M. A. Batalin and G. S. Sukhatme, "Efficient Exploration without Localization," *Intl. Conference on Robotics and Automation* (*ICRA*), pp. 2714–2719, Taipei, Tanwan, May 2003.

[4] M. A. Batalin and G. S. Sukhatme, "Coverage, Exploration and Deployment by a Mobile Robot and Communication Network," *Proceedings of the International Workshop on Information Processing in Sensor Networks*, pp. 376-391, Palo Alto Research Center (PARC), Palo Alto, Apr 2003.

[5] J. Hill and D. Culler, "A Wireless Embedded Sensor Architecture for System-level Optimization," *Technical report*, Computer Science Department, University of California at Berkeley, 2002.