ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 25 (2016) 364 – 371

Procedia Technology

Global Colloquium in Recent Advancement and Effectual Researches in Engineering, Science and Technology (RAEREST 2016)

Obstacle Avoidance in Mobile Robotic Sensors and Establishing Connection

Sherin Ann Sebi[a], Divya Sunny*

[a]*M.Ttech sudent, St.Joseph's College of Engineering and Technology,Palai, Kottayam, India*
[b] *Asst. Professor, St.Joseph's College of Engineering and Technology,Palai, Kottayam, India*

Abstract

The Mobile Robotic Sensor Network (MRSN), require reliable applications and coordination based on strong network connectivity. These sensors are deployed into hazardous environment such as extreme temperature, mudslides, explosion etc. Chances of failure is high in sensor during operation which leads to easy compromise on connectivity due to interference. Based on the survey undertaken on DARA, LeDiR, they faces obstacle collision and sensor collision. The proposed method OCRS avoids the problem of obstacle avoidance and also introduces the concept of gyroscopic force. OCRS is implemented with the help of backup selection algorithm. On the basis of implementation done with the help of system model and control input it proves OCRS is more efficient than the existing system.

*Keywords:*Mobile robotic sensor network; connectivity restoration; MANET; obstacle avoidance;

1. Introduction

Wireless Sensor Network (WSN) is also known as Wireless Sensor and Actor Network (WSAN). It is a specially defined independent or autonomous sensor [1]. WSN is initially used in military cases for monitoring as well as reporting. From this motivation, the WSN is used in everywhere for all means. The WSN or WSAN it have a node, which is named as sensor node [2]. The node is connected to tens or hundreds or thousands of several other sensor nodes. A sensor node consist of three or four components, such as radio transceiver, transducer, micro-computer or micro-controller, and a power supply [3]. The transducer provide the signals i.e., the radio signals. The radio transceiver will receive the signals as the input or as data. The micro-computer will store the data and monitor it, hence the name micro-controller. Power supply is obtained from battery. So battery is considered as a power supply in WSN

doi:10.1016/j.protcy.2016.08.119

or WSAN [4].

WSN or WSAN are of different types. MANET, VANET, SPAN, iMANET, etc. are few among them [5]. MANET is Mobile Ad Hoc Network which deals with mobile network and communication. VANET is Vehicular Ad Hoc Network which deals with vehicular network and the way of communication in between vehicles as well as with obstacles. SPAN is Smart Phone Ad Hoc Network which deals with smart phones and its communication network. iMANET is Internet-MANET. All these types follow the properties or characteristics of WSN or WSAN. Scalability or ability to move, ability to cope with heterogeneous network, resilience, availability, flexibility, faster communication etc. are its features.

There are various applications in this field. Area monitoring, industrial monitoring are few among the application categories [6]. In case of area monitoring, it deals with, air pollution monitoring, forest fire monitoring, environmental/earth monitoring etc. The earth/ environmental monitoring deals with land slid detection, water/waste water management monitoring, pollution detection etc. In case of industrial monitoring, it deals with, data logging, data management monitoring etc [7].

The outline of the paper is as follows: Section 2 presents related works. Section 3 presents the proposed work and section4 represents the implementation. Finally section 5 concludes the paper.

2. Related Works

2.1. Distributed actor recovery algorithm 1-c

Distributed Actor Recovery Algorithm, DARA, [8] – [9] is actually does recovering from a node failure in wireless sensor – actor networks with minimal topology changes. There are two types of DARA. One is DARA 1 – C and the other one is DARA 2 – C. This is based on the number of connectivity. DARA 1 – C initiates the recovery within the neighbourhood for the failure node. Following are the steps of DARA 1 – C:Heartbeats and neighbour list maintenance, Detecting actor failure and initiating the recovery process, Best candidate selection, Cascaded node relocation. The initial step will be Heartbeat and neighbour list maintenance, where each node should collects its neighbour information, usually the one-hop neighbour. Here each node will act as an actor. As each time the position of the actor changes, the list should be updated. The neighbour list is stored in a table, it consist of three parameters: { node degree, position, ID }. Nodes will periodically send heart beat messages, this is to ensure that they are still functional Detecting Actor Failure and Initiating the Recovery Process. Here the missing heart beat messages leads to the detection of failure of actors. Based on the position of actor in the network, no recovery or a major recovery will be done. Whenever cut – vertex node fails, DARA 1 – C focuses on the restoration process in the inter-actor connectivity. The next step is the Best Candidate Selection. By substituting Af, failed actor, with one – hop neighbours, the connectivity of the partitioned network is restored. But the problem will be which neighbour should be selected. In order to identify the best candidate, they should undergo three criteria: Least Node Degree, Closest Proximity to Failed Nodes, Highest Actor Degree. The final step is the Candidate Node Relocation. Here the best candidate move to the failed node location and also calculate the expected time to reach to location. But before moving to the new location they send 'MOVING' message. Upon reach the new location, the best candidate will broadcast a 'RECOVERY' message.

2.2. Distributed actor recovery algorithm 2-c

In case of DARA 2 – C [8] – [9], it focuses on restoring two – connectivity after the failure of a node. When critical node fails, until the network connectivity is restored, some nodes will be temporarily isolated. Each node in DARA 2 – C maintains at least two neighbours. DARA 2 – C follows three steps and are: Detecting Failed Nodes, Selecting Node for Reposition, Node Relocation. In case of detecting failed nodes, DARA 2 – C do the same procedure as that of DARA 1 – C, i.e., they periodically send heart beat messages to the neighbouring nodes. Missing heart beat messages leads to the detection of failed nodes or actors. For selecting node for reposition, they have to follow three criteria as in case of DARA 1 – C and are: Lowest Node Degree, Least Distance, Highest Actor ID. The final step used in DARA 2 – C is Node Relocation. The best actor will notify to all its neighbours that it moving, there by confirming the neighbour the time it takes to reposition and also the location it is intended to reposition. When the best candidate moves its position, they broadcast a 'RECOVERED' message to its dependents, indicating the completion of the process.

2.3. Least disruptive topology repair

Later introduced the Least Disruptive Topology Repair, LeDiR method [10]. This process focuses on partial view of network topology via a routing table [11]. This method follows four steps of restoration process and are: Failure Detection, Smallest Block Identification, Replacing Fault Node, Children Movement. The first three methods are same as that of DARA. The final step deals with movement of the children. Suppose if node J moves to the failed node location to restore connectivity, it possible that some of its children may fail to get direct connection or link to it. In general, it shouldn't happen when some data or paths are extended. LeDiRopts to avoid this problem and helps to sustain the existing links. If a child receives a moving message from its parent P, the child passes this information to its neighbours i.e., the grandchildren of P and moves to the direction of new location of P, until it get reconnected again with parent P. Suppose if a child receives notification from multiple parents, the it move towards the parents and selects a new location in which it is feasible to get connected with the two parents. Suppose if child C have two parents A and B, who move towards the previous location of node J, where node J has moved to the faulty node location since node J is the best candidate. So node A and B moves to the previous location of node J until they reaches half the radius units away. Before moving, these parent node inform it to the child node C. As a result node C moves to the new location of nodes A and B and find a suitable slot and occupy the new location of node C in which it is still connected to nodes A and B. Ideally, the child node C will move toward the closest point that lies within the closest communication range of A and B, this is the closest intersection point of the two circles of radius 'r' and centred at node A and node B.

The aforementioned methods neglect the important problem of obstacle avoidance. The online obstacle avoidance refers to the reactive approach that does not require the knowledge of obstacles in advance [11]. A feasible way to avoid local minima problem is to use the gyroscopic force, to handle obstacle avoidance in addition to potential functions [12]. Most mobile robotic sensor networks may be deployed in hazard environment, and these are subjected to potential risks. These potential risks include extreme temperature, explosion, landslides and mud slides, hurricane etc.

3. Proposed Work

3.1. System Model

Considering the system model of a mobile robotic sensor, it have M mobile robotic sensor. Encode the inter sensor network, in terms of undirected graph, G. Graph consist of sets of edges and vertices. It also uses the concept of Euclidean distance. These all are allocated in a region, R. so this network model deals with the following definitions and are Connectivity, 1 – Hop and 2 – Hop neighbours, Collisions. In case of connectivity, an undirected graph is connected, provided there must be at least one path in between any two distinctive vertices. In case of 1-hop and 2-hop neighbours, j is the 1-hop neighbour of I and Ni is the 1-hop neighbour set of j. Similarly, k is the 2-hop neighbour of i and square of Ni is the 2-hop neighbour set of i. there is a sensing shell with a circle of radius Rs, and is centred at each sensors. As a result, the sensor can respond to any obstacles within it. In case of collision, a collision shell with radius Rc is centred at each robotic sensors, such that the collision occurs when a sensors collision shell intersect with the collision shell of the another sensor. The main objective is that a good backup selection algorithm i.e., backup selection for cut-vertex and also restore connectivity with collision avoidance. For this restoration process, design a controller to drive the backup sensor towards the locations of the failed sensor node and also to avoid any collision between any sensors and obstacles.

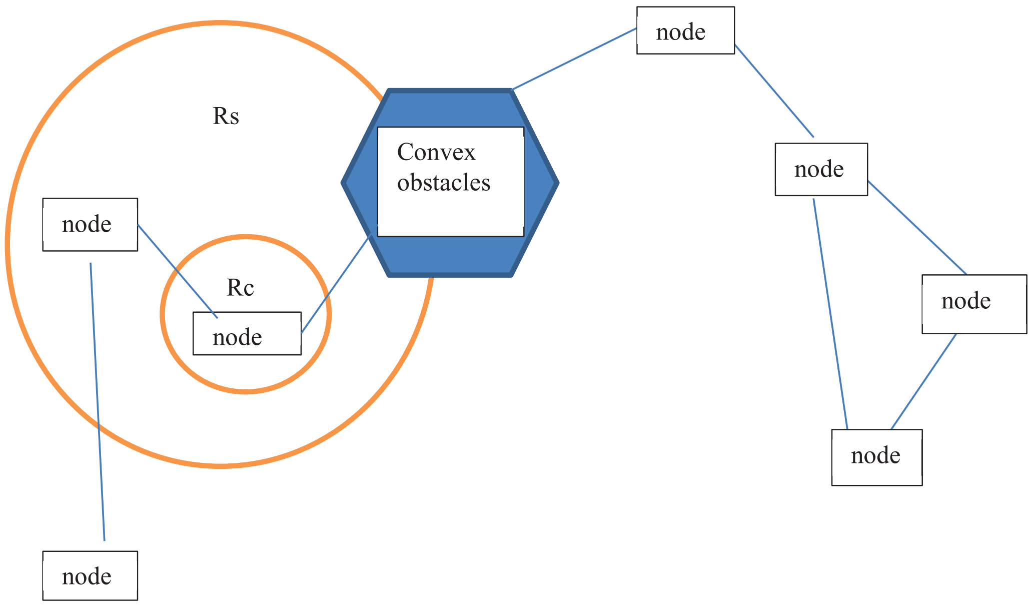

Fig. 1. Illustration of network, sensor and obstacle model

3.2. Backup Selection Algorithm

The backup selection algorithm deals with three steps and are:

- Initialization
- Determination
- Backup Selection.

In case of initialization, by periodically sending heartbeat messages to the 1-hop neighbour, as a result the sensor node will collect the neighbour's information. Here the heartbeat message consist of three parameters and are sensors ID, geographic position and the one hop neighbour set, Ni. After successfully collecting the information, each sensor can build a complete set of 1-hop neighbours and a complete set of 2-hop neighbours. In case of determination, based on the complete set of 1-hop and 2-hop neighbour set, a sensor node, suppose sensor I, can determine whether there exists a cut vertex or not in the graph. For this purpose, consider a spanning sub graph which consist of set of 1-hop neighbours and sets of 2-hop neighbours. A Laplacian matrix [14] is used, in which, it is the difference between a diagonal matrix and an adjacent matrix. The resultant will be matrix, i.e., it's a symmetric positive semi-definite matrix, and all its eigen values will be non-negative. The eigen vector corresponding to the first eigen value is always one, 1. The second eigen value is the algebraic connectivity of the system and should be greater than zero [15]. So if it is greater than zero then it is a non-cut-vertex, otherwise it's a cut-vertex. Now consider the next step, i.e., the backup selection, whenever a cut-vertex is identified, then a possible backup node should be used or positioned on the failed node position. The backup node is selected from the set of 1-hoop neighbours and then follows three steps. First, find the total degree of each node of possible cut-vertex. The total degree is identified in a way such that, it takes the union of set of 1-hop and 2-hop neighbours. Among this, we get an output set in which, for each element in the set will be the total degree of each node. Now the possible cut vertex is determined as sensor i. So secondly, the union of total degree will be send as a request message to the sensor node i. Finally from this set the least total degree will be selected and is considered as the backup sensor. The advantages of using total degree is that, a smaller group of sensors is required to move during the restoration. This will effectively reduce the travel distance as well as energy consumption. BSA allows sensors to back up a node for multiple cut-vertices. So it guarantees that every cut vertex sensors can locate a backup among its 1-hop neighbours regardless of the network topology density.

3.3. Obstacle avoidance connectivity restoration strategy

An online obstacle avoidance connectivity restoration strategy, OCRS, is used to completely explore the mobility of the sensor to restore the network connectivity. At first, OCRS use only local information of 2-hop, to determine the cut-vertex. The determination is based on the graph Laplacian method [13]. Then, a backup selection algorithm is used. At last, a gyroscopic force based motion controller is used for obstacle avoidance. Vectors have direction, momentum and force. Torque is the product of radius and force applied. It is always perpendicular to the direction of force. Gyroscopic force can be explained with the help of a bicycle wheel. Suppose a bicycle wheel is hanged in a rope and is in way that the wheel is parallel to the surface of earth. If we rotate the wheel and leave it, it just rotate facing to the surface of the earth. If we hang the wheel perpendicular to the surface of the earth and if we rotate the wheel and leave it, it will move with an angle of above sixty degree. This is due to angular momentum and hence it is known as gyroscopic force i.e., this leads to gyroscopic precession.

Considering the case of gyroscopic force, the control input will be the sum of different types of forces. The forces are target potential force, attractive force, gyroscopic force and a breaking force. In case of restoration, the backup sensors will drove it to the position of the failed sensor node to undergo restoration. For obstacle avoidance connectivity restoration strategy, considering the situation of a restoration process, to move to the target location, the best candidate uses a particular force, which is the sum of breaking force and the gyroscopic force. Due to the application of this gyroscopic force, the collision of sensors or the obstacle or in between them are avoided to a large extend. Based on paper [11], the number of collisions, total number of messages, number of sensors moved, total travel distance etc. are reduced to a large extend. While comparing it with the already existing system, we could find a vast change. No collision can be observed in obstacle connectivity restoration strategy, it is due to the fact that the gyroscopic force and the breaking force will navigate the sensor away from obstacles. The increased density of a network creates more changes in the case of inter-sensor collision as well as sensor-obstacle collision. The total number of messages in obstacle connectivity restoration strategy is smaller due to its simple backup selection algorithm and it require only very limited amount of information of the network topology. OCRS seems to respond quickly or as fast as possible in a large scale mobile robotic sensor network. When the density of the network increases, then the OCRS require more sensors in the restoration procedure. The total travel distance is small compare to DARA. The total travel distance is small and is steady for OCRS with increased density of network.

4. Implementation

DARA is not able to determine the cut-vertices in the network, since it only have 1-hop neighbor information. LeDiR deals with the case of multiple node failure, where it faces a problem of timely restoration of mobile robotic sensor network. Both these methods avoids the problem of obstacle avoidance. According to OCRS, it has to generate a control input, which consist of four different forces.

$$U = F_r + F_a + F_b + F_g$$

To avoid local minima problem, the concept of gyroscopic force is introduced. This force will steers the sensors away from obstacles. The breaking force will decelerate the sensors moving to the obstacle. The attractive force will maintain the existing links and the target potential force backup the sensors to their position for connectivity. If there is no obstacles, the sensor will move with the help of target potential force and attractive force. If there exists an obstacle, then along with these two forces, a gyroscopic force and a breaking force will be added. Based on the working of these forces, the obstacles will be avoided and also best route will be adopted. Gyroscopic force works only if the nearest distance between sensor and obstacle should be less than the sensing shell. While simulating, should consider the control inputs under the scenario of with obstacles and without obstacles. For OCRS a backup selection algorithm needs to be implemented, which has been stated above along with the working. The algorithm is adopted with the help of laplacian matrix and laplacian graph concepts. The following figure2 shows the simulation of sensors moving in a mobile robotic sensor network. The nodes with red circles are the nodes that are approaching to an obstacle. So when it reaches the critical region they work on the basis of gyroscopic force as in obstacle avoiding connectivity restoration strategy. So when the sensor node gets out of the region, the red circle will be removed. OCRS provide less collision, less travel distance and also less number of messages. The following figure3 shows the graphical representation of number of collisions and also the total travel distance, based on without OCRS and with OCRS.

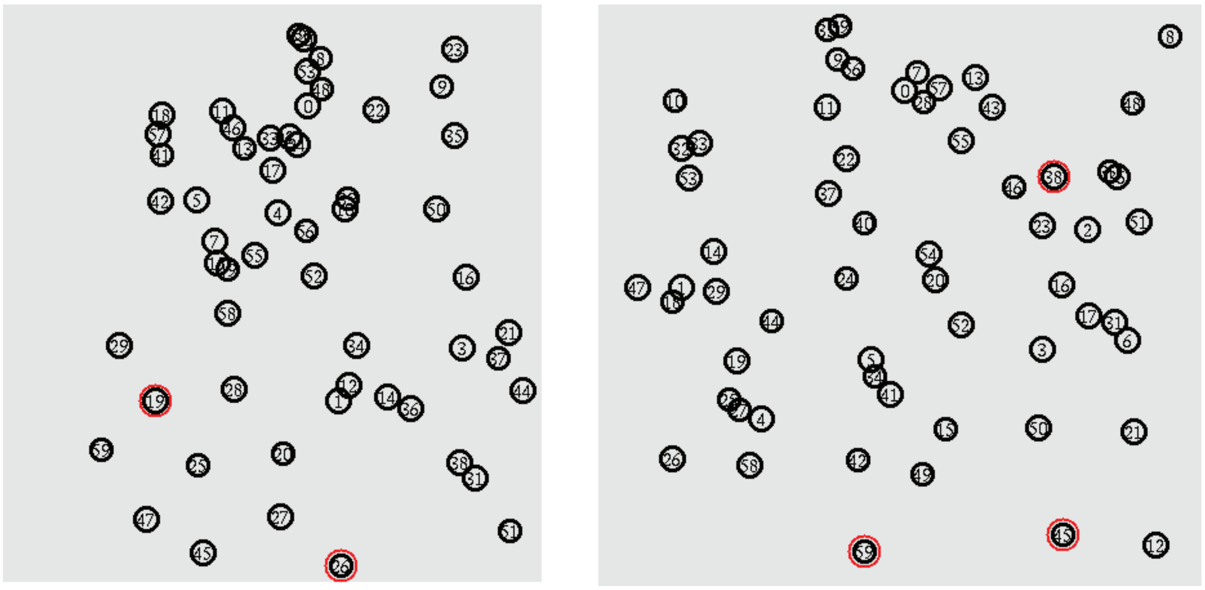

Fig2: (a) & (b) Node movement in the sensor network

The modules used in the implementation of OCRS are the network model, connectivity, one hop and two hop neighbour, collision shell, backup selection for vertices, sensor displacement, restore connectivity, OCRS. In network model, have to provide the sensor nodes that are mobile and also should provide the routing mechanism. This will be done inside the Mobility file. Initially nodes will be initialized and declared in the Setdest file. Obstacles are also introduced in the same manner as mobile nodes but along with it obstacle functions are called from Obstacle file inside the Mobility folder. Connectivity is obtained with the help of routing protocol. One hop and two hop neighbour are introduced in AODV file, with the required set of equations and conditions to obtain the information, and also collision shell is introduced in the AODV file, where the radius of the sensors will be considered to identify the shell. The mechanisms will be provided in the aodv file. The backup selection for vertices is a module that actually does the backup selection algorithm. This is introduced in the same AODV file provided this module have three sub module and are: initialization, determination, backup selection. Sensor displacement module and restore connectivity module are declared in Mobility file, but are defined in the AODV file. The OCRS module is implemented in the AODV file, where the control inputs will be declared and based on the actions of the forces, obstacle avoidance conditions will be provided. Based on these set of conditions, checks the situations and the process to avoid obstacles is implemented. The above figure shows the simulation output, just the animation.

5. Result

In the following figure, (a) shows the result of number of collisions with OCRS and without OCRS. The X-axis of the graph shows the number of sensors and the Y-axis shows the number of collisions. Considering the graph obtained we can see that even if the number of sensors varies, the collision rate is very slowly increasing, whereas in case of without OCRS, we could see some steep growth. The collision rate is high in case of without OCRS. An example is that when looking at OCRS, when the number of sensors is 20,000, the number of collided nodes will be approximately 37. When the number of sensors is high, of about 80,000 to the collided nodes are less and are about approximately 51. While considering the scenario of without OCRS, when the number of sensors are 20,000 from figure, the collision rate is high and is 59, and when the sensors are about 80,000, the collision rate will be 80. Now considering figure (b), the X- axis deals with number of sensors and the Y- axis deals with the total travel distance. In OCRS, even if the number of sensors changes in large numbers, the travel distance will be increasing very slowly. But in case of the situation of no OCRS, the travel distance is huge foe each set of sensors, which is evident from the graph. A good example is that , while looking at the graph, when the number of sensor nodes are 20,000, the distance occupied will be approximately about 150 for OCRS, whereas in case of without OCRS, for the same number of sensor nodes, the travel distance actually starts from 50, which seems to be huge.

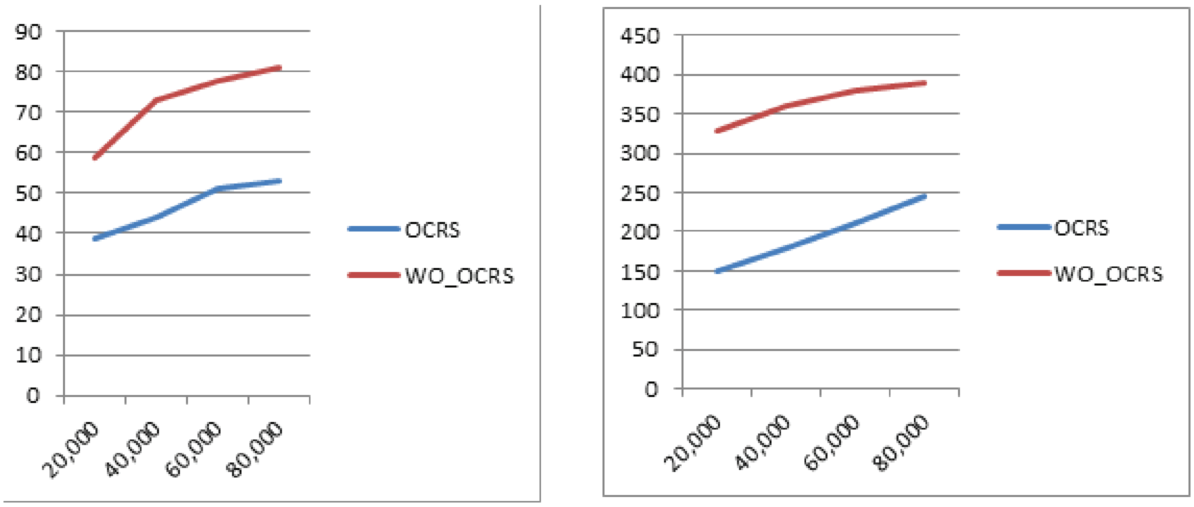

Fig3: (a) Number of collision, (b) Total travel distance

6. Conclusion

This paper investigates on the importance of connectivity and restoration on mobile robotic sensors that works under harsh environments. The existing systems avoid the problem of obstacle avoidance. The paper suggests an effective methodology named, online connectivity restoration strategy, which withstands under real world working conditions. A theoretical study is undergone to compare the efficiency of OCRS and is proved based on the successful implementation and the results obtained. OCRS is more efficient, provide less time complexity and also avoid or reduce collision of nodes.

References

[1]http://searchdatacenter.techtarget.com/definition/sensor-network

[2] https://en.wikipedia.org/wiki/Wireless_sensor_network

[3] H. F. Rashvand, A. Abedi, J. M. Alcaraz-Calero, P. D. Mitchell, and S. C. Mukhopadhyay, "Wireless sensor systems for space and extreme environments: A review," *IEEE Sensors J.*, vol. 14, no. 11, pp. 3955–3970, Nov. 2014.

[4] Q. Liang, X. Cheng, S. C. Huang, and D. Chen, "Opportunistic sensing in wireless sensor networks: Theory and application," *IEEE Trans. Comput.*, vol. 63, no. 8, pp. 2002–2010, Aug. 2014.

[5]http://techterms.com/definition/manet

[6] https://en.wikipedia.org/wiki/Mobile_ad_hoc_network

[7] https://datatracker.ietf.org/wg/manet/charter/

[8] Abbasi, M. Younis, and K. Akkaya, "Movement-assisted connectivity restoration in wireless sensor and actor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 9, pp. 1366–1379, Sep. 2009.

[9] Z. Mi, Y. Yang, R. Yu, and H. Ding, "Distributed connectivity restoration for mobile sensor systems with limited information," *IEEE Sensors J.*, vol. 14, no. 11, pp. 3838–3850, Nov. 2014.

[10] A. Abbasi, M. F. Younis, and U. A. Baroudi, " Recovering from a node failure in wireless sensor- actor networks with minimal topology changes," *IEEE Trans. Veh. Technol.*, vol. 62, no. 1, pp. 256–271, Jan. 2013.

[11] ZhenqiangMi, *Member, IEEE*, Yang Yang, and James Yifei Yang, "Restoring Connectivity of Mobile Robotic Sensor Networks While Avoiding Obstacles", IEEE Sensor J., vol. 15, no. 8, Aug 2015.

[12] D. E. Chang, S. C. Shadden, J. E. Marsden, and R. Olfati- Saber, "Collision avoidance for multiple agent systems," in *Proc. 42nd IEEE Conf. Decision Control*, vol. 1, Dec. 2003, pp. 539–543.

[13] C. D. Godsil, G. F. Royle, and C. D. Godsil, *Algebraic Graph Theory*, vol. 207. New York, NY, USA: Springer- Verlag, 2001.

[14] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state- dependent graph Laplacian," *IEEE Trans. Autom. Control*, vol. 51, no. 1, pp. 116–120, Jan. 2006.

[15] A. A. Abbasi, M. Younis, and K. Akkaya, "Movement- assisted connectivity restoration in wireless sensor and actor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 9, pp. 1366–1379, Sep. 2009.