Router Node Placement in Distributed Sensor Networks: A Review of Optimized Methods

Abhishek M. Kinhekar, Parmalik Kumar

Abstract*— The recent advancements in Distributed Wireless Sensor Network has stimulated the need for the newer and enhanced version of algorithms, which will not only reduce the delay in the processing but also consumes much less power. Distributed Sensor networks are most employed and have much scope for their optimization in working. In this paper we explore to find and compare about wireless sensor network, router placement, synthesis algorithm and simulation tools of DWSN.

Index Terms*— wireless sensor network; router placement; synthesis algorithm; Simulation Tools

I. INTRODUCTION

Distributed Wireless Sensor Networks (DWSN) has its applications in many real time areas like modelling, health care, defence, environmental monitoring, security and many more. One of the most significant applications of WSN is in building Automation Systems (BAS). As the designing of a complicated WSN for BAS is not an easy manual task, a Computer Aided Design (CAD) tool or a simulation tool plays a significant role. A wireless sensor network consists of a number of wireless sensor nodes. These nodes are characterized by being very small in size with limited energy usually supplied by a battery. They communicate via built-in antennae over RF signals. These networks are typically used to monitor a field of interest to detect movement, temperature changes, precipitation, etc. One of the most active research fields in wireless sensor networks is that of coverage. Coverage is usually interpreted as how well a sensor network will monitor a field of interest. It can be thought of as a measure of quality of service. Coverage can be measured in different ways depending on the application.

In addition to coverage it is important for a sensor network to maintain connectivity. Connectivity can be defined as the ability of the sensor nodes to reach the data sink. If there is no Available route from a sensor node to the data sink then the data collected by that node cannot be processed. Each node has a communication range which defines the area in which another node can be located in order to receive data. This is separate from the sensing range which defines the area a node can observe. The two ranges may be equal but are often different. There are several factors that must be considered when developing a plan for coverage in a sensor networks. Many of these will be dependent upon the particular application that is being addressed. The capabilities of the sensor nodes that are being used must also be considered. Most researchers focus on a single deployment model but there are papers that attempt to develop a more general algorithm that can be used in many types of deployment.

Revised Version Manuscript Received on May 26, 2015.

Abhishek M. Kinhekar, B.E. Department of Computer Science Technology from Nagpur University. Guru Nanank College of Engineering, Nagpur. India.

Prof. Parmalik Kumar, M. Tech. Department of Computer and Information Technology. RGPV. Bhopal, India.

II. BACKGROUND

This template, modified in MS Word 2007 and saved as a "Word 97-2003 Document" for the PC, provides authors with most of the formatting specifications needed for preparing electronic versions of their papers. All standard paper components have been specified for three reasons: (1) ease of use when formatting individual papers, (2) automatic compliance to electronic requirements that facilitate the concurrent or later production of electronic products, and (3) conformity of style throughout a conference proceedings. Margins, column widths, line spacing, and type styles are built-in; examples of the type styles are provided throughout this document and are identified in italic type, within parentheses, following the example. Some components, such as multi-leveled equations, graphics, and tables are not prescribed, although the various table text styles are provided. The formatter will need to create these components, incorporating the applicable criteria that follow.

III. BACKGROUND

Related Work

In [1, 2, 3] authors illustrated tools and methodologies for the modelling, simulation and automatic code generation of WSN applications. In this paper, we intend to discuss methods and their implementations to overcome these issues or rather we propose better solutions to designing distributed networks. The authors of [11] present a sophisticated design tool that is able to assist a designer in designing WSNs. They argue that it is possible to design a network that is more resilient to failure and has a longer life time. To achieve that they add redundancy to the network which will increase resiliency and place routers only where they are needed to improve life time. Both of these have a direct relation to router placement. To synthesize a network they propose two methods of synthesis, one that yields and exact solution to the problem and a heuristic that results in a sub-optimal solution. The exact method employs the Mixed Integer Linear Programming (MILP) optimization. In relation to our work, we are using a heuristic method to solve the issue of router placement. In Gibney et al. [2], their article present a tool to assist in the design of a Building Management System. This tool first gathers specific requirements such as the target environment constraint, measurements zones and building geometry. They then discuss the methods of generating candidate sensor position using the Neural-Gas algorithm. Using this algorithm, the network topology is incrementally generated. Site specific demand zones are identified and this information is passed on to the algorithm. Based on this information the Neural-Gas algorithm will be able to generate the candidate positions. After this an agent based optimization I used to

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

optimize the network that was synthesized. In contrast to our work, this paper finds the candidate position of sensors and not routers. Our work assumes that sensor positions are fixed and only router positions are at play. Wang et al. [3] target sensor deployment in indoor environments. Their objective is to develop a more effective way to deploy sensor networks and minimize the number of nodes deployed while guaranteeing coverage and connectivity. They propose using several search algorithms such as Simulated Annealing. Their deployment method consists of partitioning the sensing field into smaller sub-regions based on its shape and then deploying the sensors. We do not partition the sensing field or the network 3 floorplan. We consider the target space as a whole and we are only concerned with the positioning of routers not sensors. In their research, Pinto et al. [4] present a design tool to automatically design a network. The synthesis algorithm presented in their work employs the use of MILP which would yield an exact solution. We differ from this work as our work does not use any Integer Linear Programming (ILP) to result in an exact solution. We are using a heuristic method to synthesize the network.

In [4, 5], authors illustrated tools and methodologies for the modeling, simulation and automatic code generation of WSN applications. In this paper, we extend existing CAD tools by proposing a tool for network synthesis. In particular, we introduce an interactive tool that optimizes network topology, i.e. the location of router nodes. The tool facilitates users by reducing design time and by improving the quality of the network topology in comparison to a simulation-based approach, where designers have to simulate several different topologies to get the most qualified one, with no guarantee of optimality. Users can interact with the tool through a GUI, and add new information about the network behavior and the deployment environment, according to their fields of expertise. The tool takes this information into account to incrementally adapt the node positions, and it provides feedback to the designers by analyzing network performance.

Expectations from the Synthesized and optimized Algorithm

In[1] paper, authors proposed robust and efficient synthesis algorithms for optimal node placement for indoor environments. With the proposed algorithm, authors were able to expect over 1000% improvement in execution time and over 150% improvement in reducing the number of placed routers. the proposed synthesis algorithms produced in [1] has better network architecture compared to existing grid based simplistic network. Networks are prone to failure. This could be due to node failure, power failure or even natural hazards. To overcome this, network should have some amount of redundancy.

IV. METHODOLOGY

Majority of mentioned works is based on the placement of sensors. In contrast, we assume that sensors are placed and sensing coverage is adequate; thus we are primarily concerned about the deployment of routers to automatically synthesize the network. Authors in [1][3] proposed that the tool (NS2) is composed of a Graphical User Interface (GUI) as the front-end and synthesis algorithms as the back-ends of the tool. The entire design starts by running the GUI. The GUI asks the user for total no of routers, area covered by each router, and calculates the priliminary solutions on the basis of algorithm. The tool then asks for the optimised algorithm and calculates synthesis process. The goal of the combination is to find candidate positions to place routers.

Author's refered to nodes placed by the user on the GUI as physical nodes and the nodes placed by the synthesis algorithm as virtual routers (VR). In first steps of the synthesis algorithm, the number of end devices which don't have a path to BS is counted, stored in an iteration number. Afterward, candidate coordinates of virtual routers are generated based on the value of iteration number. In fact, the virtual routers are placed in a rectangular grid fashion according to the algorithm I. In the next step, the corresponding lines between these virtual routers and the physical nodes are retrieved. The signal strengths can be calculated according to these lines lengths and the obstacles perceived by the pixel values of these lines. Then, the proper candidates based on minimum signal strength are chosen to be passed to Dijkstras's algorithm. After adding the virtual nodes, the algorithm examines the paths considering the whole network to make sure all the nodes are connected to the BS. The previous procedures will continue until all of the end devices are connected to the BS. the proposed algorithm in [1] is as under Algorithm I :

Generating candidate coordinates Input: iteration

if iteration = 1 then

for k=0 to 2 do

for j=0 to 2 do

[x,y] = [5+k*middleLength, 5+j*middleWidth]

newNode(x,y)

else

for k=1 in steps of 2 to 2 ^ iter do

for j=0 to 2^ıter do

[x,y] = [5+k*middleLength, 5+j*middleWidth]

newNode(x,y)

for k=0 in steps of 2 to 2 ^ iter do

for j=1 in steps of 2 to 2 ^ iter do

[x,y] = [5+k*middleLength, 5+j*middleWidth]

newNode(x,y)

A. Modification for Node placement

As per [3] the reference synthesis algorithm for router placement does not take into account the geometry of the floor plan and does not optimize important metrics like the number of placed nodes which affect final design cost. This algorithm assumes the nodes according to the reference method which are placed in rectangular grid fashion. To solve this lacuna, the network based on our algorithm is synthesized with routers which are placed in their maximum communication radius and in straight lines between the base station and the end-devices. The algorithm for placing nodes is presented in algorithm II.

Algorithm II : Main synthesis algorithm

Input: User input of end-devices & base station (BS) locations

for k=1 to of physical nodes excluding BS do

retrieve x-y coordinate of BS and physical node (determined by loop index) in feet

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

convert x-y coordinates to pixel x-y coordinates

generate line equation from BS to physical node

collect points on the line in small steps

prune all unique points

for j=1 to of unique points do

if point value = 0 then

calculate signal strength from BS to that point

numberOfObstacles ++

[flag] = placeRouter(unique points, signal strength)

if flag = 1 then

exploreNew(unique points)

else

calculate signal strength

placeRouter(unique points, signal strength)

The program initially starts the same way as the reference algorithm. After taking necessary inputs from user, the synthesis process starts. First, the algorithm retrieves the x-y pair coordinates of the base station and the physical nodes in feet. It then converts these to the corresponding x-y pixel coordinates. Afterward, it generates the line equation from the base station to the current physical nodes. Considering the binary image of the floor plan, the algorithm iterates in small steps and all points that lie on the line is collected; thus the unique points are checked for being an obstacle. Iterating in small steps ensures that there is no considerable gap in the line and we do not lose any probable obstacle. Having the number of obstacles by checking the pixels, the signal strength is calculated. This calculated value is then compared with the minimum signal strength according to algorithm III and the corresponding node as a router is added to the network.

Algorithm III : Comparing signal strength and placing router

Input: unique points, signal strength

if signal strength $\leq$ *minimumsignalstrength* then

if point value = 0 //obstacle then

reiterate backwards to first pixel that is not an obstacle

move another 15 points back // that is a new point

convert new point to feet

if new point is x distance away from old point then

flag = 1

place point on G UI

draw line connecting current node to previous node

As per [1], There are a few lacunas in the mentioned algorithmt he first issue is the situation in which the candidate position is found on an obstacle. Then the algorithm will go backwards in the corresponding line to keep the distance from the obstacle to place the virtual router. The second issue is the fact that there should be enough space between the recently added virtual router nodes in order to satisfy the optimization goals. If a node is placed too close to the previous node, the algorithm should look forward for an alternative position to place the node. Thus an 'exploratory' technique is employed in order to find an alternative path, shown in algorithm IV.

Algorithm IV : Finding an alternative path

Input: unique points

Ascertain end-device position with respect to BS

Identify x and y coordinates of the end-device

for Attempt to move in direction of the x-component of the end-device do

keep track of distance and number of pixels that are obstacles

calculate signal strength

for Attempt to move in direction of the y-component of the end-device do

keep track of distance and number of pixels that are obstacles

calculate signal strength

Compare signal strength

if signal strength from moving in x-component is greater then

place router at that position else

place router in the other position

Considering the straight line between the base station and the end device as a vector, we can decompose it into x and y components. Now, we can explore regions around the non-suitable node in x and y directions. Considering obstacles in each direction, we then choose a direction for the path in which the signal strength is stronger. A router node will be placed at the maximum communication radius.

B. Modification for Node clustering

Clustering method is implemented to optimize the matric of no. of nodes. The basic idea of clustering is to connect nodes that are within each others communication radius. Conventionally, clustering starts at the end devices and moves backwards towards the BS. When two or more nodes fall into each others communication radius, they are combined into one node and all nodes that are below that node (connected from that node to BS) are removed. When one end device can connect to the neighbor's router, the paths will be modified and the extra router will be removed from the network. In order to implement the corresponding algorithm for clustering, the paths from the previous synthesis algorithms must be first saved into a data structure. Afterward, the clustering method is executed according to algorithm V. The clustering algorithm which is based on *neighborhood* check, determines if other nodes fall within the communication range of a specific node in a path. The algorithm starts with examining each of the nodes (test node) in all paths reaching the BS. During the clustering process, the nodes which are in the neighborhood of the test node and are on other paths are pruned out.

Algorithm V : Clustering algorithm - part one

Input: paths from BS to end-devices

for k=1 to # of paths do

for j= # of nodes in path k to 2 do

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

extract added nodes from data structure excluding BS

[nodes in 75dB range] = check75dBConnectivity (node x-y coordinate, added node) [reduced nodes in 75dB range] = remove nodes in range that in path k [nextNodeFlag,newPathFlag, signal strength] = lineEquation (reduced nodes in 75dB range, node x-y coordinate) if next Node Flag = 1 then continue to next node defined by index j

if next Path Flag = 1 then [modified path] = build Path (paths from BS to end

Then the test node is examined whether it can connect to those neighborhood nodes, considering obstacles. If the test node is unable to connect to other neighborhood nodes then the algorithm goes for the next node on the same path and continues the process. On the other hand, if the test node is able to connect to another node then we will use the node that is chosen by the algorithm VI. The algorithm VI chooses a neighborhood node based on how strong the signal strength is between that node and the test node. The data structure containing the paths will be updated in order to use this selected neighborhood node and remove the old path from test node to the BS.

Algorithm VI : Clustering algorithm - part two

Input: reduced nodes in 75dB range, node x-y coordinates

for k=1 to of reduced nodes in 75dB range do

retrieve line equation from current node to node k

collect unique points

check connectivity accounting for obstacles

keep node that results in strongest signal strength

Remove nodes connected from the current node to BS

connect current node to the node that resulted in strongest signal strength

C. Decreasing run time of synthesis algorithm[1]

In [2], all pixel on the line connecting candidate virtual node to every physical nodes are collected and their corresponding signal strengths are checked. This is repeated over all generated coordinates. Since the number of pixels collected and checked in each iteration runs into the thousands, it would be an extremely time consuming process. In [2], the candidate coordinates need to be generated during the algorithm execution based on the value of iteration number. In contrast, [1],[2] propose the idea of preprocessing. In [1][2] method the coordinates need to be pre-generated. Accordingly the iteration number is determined arbitrarily or based on prior simulation knowledge. The pre-processing technique separately counts the number of obstacles that lie in the line connecting a candidate coordinate to every pixel on the floor plan. For example, lets assume that one of the x-y pair generated for candidate virtual router is (20, 20) and the floor plan size is 300 by 300 pixels. Then the program will count how many obstacles are between point (20,20) and other points. This will be done 90,000 times which is the total number of pixels in the floor plan. For every x-y pair coordinates a binary file will be written. The file is named based on the pixel value of the x-y pair.

For example, a floor plan image is 200 by 200 pixels and its dimension is 50 by 50 feet. The x-y pair of interest is (20, 20) in foot, then the corresponding pixel x-y pair will be (80, 80) in pixels. The transformation of feet to pixel is equation 1 and 2:

$X - pixel = current\ X\ position\ in\ feet\ total\ length\ in\ feet / total\ length\ in\ pixels$ (1)

$Y - pixel = current\ Y\ position\ in\ feet\ total\ width\ in\ feet / total\ width\ in\ pixels$ (2)

After pre-generating the necessary data, each synthesis process will employ a look-up table technique to extract the number of obstacles and calculate the signal strength. As a result the run time of the tool by our approach will be considerably reduced. In comparison to the reference method, another improvement in our algorithm is that we converted the original JPEG image provided by the user to a binary image; then the pixel value of '1' means free space and the pixel value of '0' means an obstacle.

V. EVALUATION

Proposed synthesis algorithms in [1] are tested on 2D floor plans with multiple obstacles. authors simulated using the old and new algorithms for the same set of nodes. Then the synthesis time of the network is logged. In addition, for every simulation the average Receive Signal Strength Indication (RSSI) and the average distance for connected nodes are computed. If a node makes three connections, then the signal strengths for each connection are added and the average RSSI is calculated. In the simulation, it was assumed that a high dense floor plan including 18 end-devices and 1 base station. Then we start our algorithm with new node placement method. The resulting synthesized network contains in total 41 nodes, among them 22 are newly placed routers. Afterward, the clustering algorithm is executed which decreases the total nodes to 28, among them 9 are newly placed routers. This procedure takes 32.78 seconds and the final network setup is shown in figure 2.

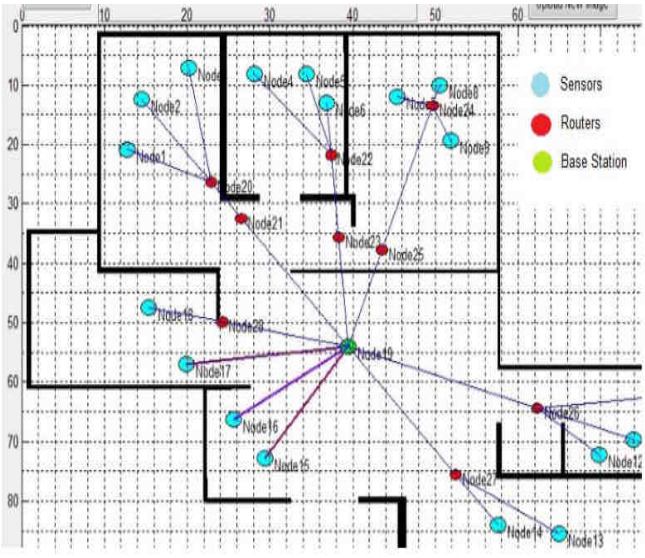

Fig 1: Final network setup based on proposed node placement and clustering algorithm

In figure 2, authors of [1] has provide run time for synthesis algorithms for seven test network setups with different

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

number of physical nodes. Run time is measured and plotted both for reference and proposed algorithms. In all network setups, the same physical floor plan size is used and the base station is placed in the middle of the floor plan, except for one simulation (simulation with 33 nodes in which the base station is placed in the upper region of the floor plan). The trend that can be seen is that the run time required to synthesize a network is less when using the new modified algorithm, compared to the reference algorithm. When running the simulation with 33 physical nodes, there were walls surrounding the base station in close proximity. This causes the proposed synthesis algorithm to process a lot more data to find proper candidate positions for routers which caused the spike in run time for that data point. Figure 3 shows the number of routers added to the synthesized network for the same test setup. This plot shows a very clean trend that the new modified algorithm always synthesizes a network with a less number of routers.

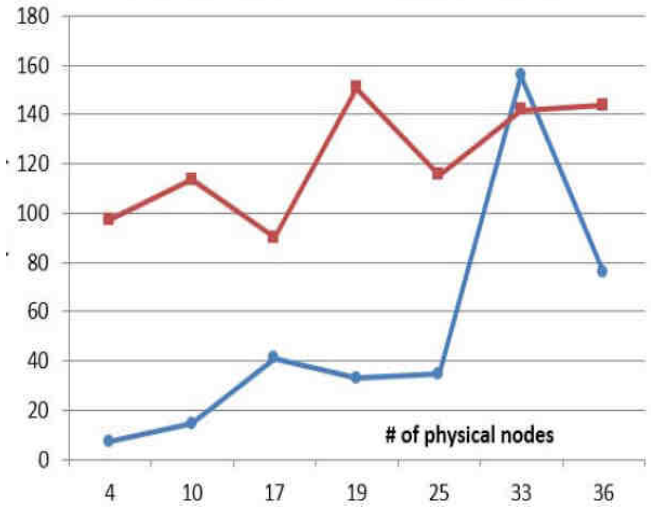

Fig. 2. Comparing run time between two synthesis algorithms

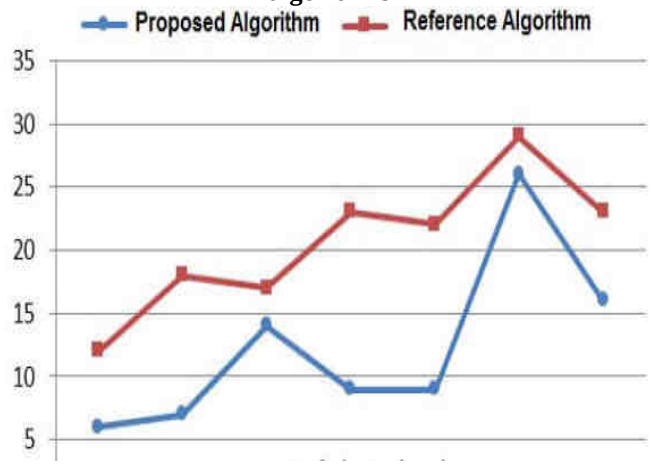

Fig. 3. Comparing number of added routers for two synthesis algorithms

VI. CONCLUSION

In this Paper, we have analyzed the basics of distributed wireless sensor networks, and presented the review of algorithms to optimize the connectivity and coverage problems, synthesis of optimization of algorithms, methodology. As future work, we plan to validate the proposed algorithm by deploying the network whose design was used as an example in the paper. The placement of the router nodes will be on randomized basis and we study the performance of different performance parameters. We are sure that measurements on network resiliency and lifetime will allow us to further tune the synthesis strategies.

References

[1] Mohammad Mozumdar, Arun Ganesan, Alireza Ameri Daragheh, "Optimizing router node placement for desining Distributed Sensor networks" 2014 IEEE International Conference on Distributed Computing in Sensor Systems

[2] A. Puggelli, M. Mozumdar, L. Lavagno, A. L. Sangiovanni-Vincentelli, "Routing-Aware Design of Indoor Wireless Sensor Network Using an Interactive Tool", IEEE Systems Journal, Volume:PP, Issue:99, 2013, pp. 1-14.

[3] Chih-Yung Chang, Jang-Ping Sheu, Senior Member, IEEE, Yu-Chieh Chen, and Sheng-Wen Chang, "An Obstacle-Free and Power-Efficient Deployment algorithm for wireless sensor networks", IEEE Transactions on Systems, Man, and Cybernetics—part a: Systems and Humans, Vol. 39, No. 4, July 2009, 795

[4] A. M. Gibney, M. Klepal, J. T. ODonnell, "Design of Underlying Network Infrastructure of Smart Building, Intelligent Environments", 2008 IET 4th International Conference.

[5] Y. Wang, C. Hu, Y. Tseng, "Efficient Deployment Algorithms for Ensuring Coverage and Connectivity of Wireless Sensor Networks", in Proceedings of First International Conference on Wireless Internet

[6] M. Mozumdar, F. Gregoretti, L. Lavagno, L. Vanzago, andS. Olivieri, "A Framework for Modeling, Simulation and Automatic Code Generation of Sensor Network Application", Proc. of SECON, 2008, pp. 515–522.

[7] M. Mozumdar, L. Lavagno, L. Vanzago, and Alberto L.Sangiovanni-Vincentelli. HILAC: A framework for Hardware In the Loop simulation and multi-platform Automatic Code Generation of WSN Applications", In Proc. of SIES, pages 88-97, Italy, 2010.

[8] Pavlos Papageorgiou, "Literature Survey on Wireless Sensor Networks", pavlos@eng.umd.edu, July 16, 2003

[9] Xuesong Liu, Burcu Akinci, and James H. Garrett, Ömer Akin, "Requirements for a computerized approach to plan sensor placement in the HVAC systems", Nottingham University Press Proceedings of the International Conference on Computing in Civil and Building Engineering W Tizani (Editor)

[10] Shaimaa M. Mohamed, Haitham S. Hamza, Imane A. Saroit, "Harmony Search-based K-Coverage Enhancement in Wireless Sensor Networks", World Academy of Science, Engineering and Technology International Journal of Computer, Control, Quantum and Information Engineering Vol:9, No:1, 2015

[11] A. Puggelli, M. M. R. Mozumdar, L. Lavagno, and A. L. Sangiovanni-Vincentelli, "Routing-aware design of indoor wireless sensor networks using an interactive tool." IEEE Systems Journal vol. PP, Issue: 99, 03 Dec 2013.

[12] A. M. Gibney, M. Klepal, and J. T. O'Donnell, "Design of underlying networkinfrastructure of smart building," in Proc. 4th Int. Conf. on Intelligent Environments, 2008, PP. 1-4.

[13] Y. Wang, C. Hu, and Y. Tseng, "Efficient deployment algorithms for ensuring coverage and connectivity of wireless sensor networks," in Proc. 1st Int. Conf. on Wireless Internet, 2005, PP. 114-121.

[14] A. Pinto, M. D'Angelo, C. Fishione, E. Scholte, A. Sangiovanni-Vicentelli. "Synthesis of embedded networks for building automation and control," in Proc. American Control Conference, 2008, PP. 920-925 .

Abhishek M. Kinhekar received the B.E. degree in Computer Science Technology from Nagpur University in 2008. He is persuing M.E. in Computer from Patel, Bhopal. He is now with Guru Nanank College of Engineering, Nagpur.

Prof. Parmalik Kumar has completed his M. tech from RGPV, Bhopal. He has vast experience in the field of Computer and Information technology. He has published several research papers.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.