Router Nodes Placement Optimization for Designing a Distributed Sensor Network

Abhishek M. Kinhekar, Parmalik Kumar

Abstract— designing a distributed wireless sensor network can be an arduous task if not done with simulation tools. Simulation with a tool should provide a robust and efficient solution of the problem with quick response time but available simulation tools for designing these wireless sensor networks are very limited. Pugelli, Mozumdar, avagno and Sangiovanni-Vicentelli[1] proposed an interactive design tool that can assist rapid design of sensor network. This tool synthesizes networks using Dijkstra's algorithm but its execution time is very high when the network size is relatively large (n>=50). Moreover, it produces sub-optimal solution with large number of router nodes. In this paper, we present efficient and robust synthesis algorithm that exclusively reduce running time.

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

I. INTRODUCTION

Distributed Recent technology advances have caused a huge expansion in Wireless Sensor Networks (WSN). Applications of WSN are not only limited to factory automation, health care monitoring, environmental monitoring, security sensing and more importantly defence application [1]. Buildings require power for lighting, heating, ventilation, air conditioning etc. The power used represents around 40% of the cumulative power consumed in the United States [1]. Looking at this percentage we can conclude that power consumed by buildings need to be controlled and monitored in an efficient manner. One attractive domain of application of WSN is in Building Automation Systems (BAS). The ability to use wireless technology in BAS gives it the edge of being able to operate in any geological position and also reduces the cost to deploy. Designing a sophisticated WSN for BAS is not an easy task to accomplish especially if done without the assistance of any simulation/prototyping Computer Aided Design (CAD) tool. In order to tackle the initial problem of not having a CAD tool to assist a designer in designing a distributed network, an interactive tool was developed in MatLab. This tool takes in a 2-D floorplan, positions of end devices such as sensors or actuators and base stations as input and then synthesizes a network of routers and the already placed end devices and base stations based on a heuristic algorithm that uses a modified version of Dijkstra's algorithm. The main goal of this tool is to synthesize a sub-optimal if not an optimal network topology (locations of the routers) [1]. The synthesized network guarantees connectivity while optimizing certain criteria such as cost, resiliency, network lifetime, etc. [1].

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.

This tool takes into account propagation models and also obstacles in order to produce a most accurate distributed network. The techniques used in this tool pose a few key issues that could possibly be an obstacle to a designer using the tool. The key issues are: 1. Rim time. For any engineer time is a precious commodity. It is not efficient for a tool to be taking a huge amount of time to produce results. 2. Node placement. The produced output is a sub-optimal solution. Node placement can be further improved for a better network topology. These issues will be further elaborated on in the succeeding chapters. In this thesis, we propose methods and implement them to overcome these issues or rather we propose better solutions to designing distributed networks.

II. BACKGROUND

Related Work

The authors of [1] 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 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 floor plan. We consider the target space as a

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

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. Chang et al. [5] discuss an effective way to develop the number of sensors required using certain search oriented strategies such as Simulated Annealing. Their work deploys nodes based on the target space and the radiation pattern of the antenna. Initially a set of fixed nodes are laced then the search algorithm goes through the search space for a solution that satisfies the coverage and connectivity constraint. In our work we initially place fixed nodes based on connectivity then cluster these nodes based on their connectivity. We do not concern ourselves with coverage as it is assumed that the sensors placed provide enough coverage. Akshay et al. [6] highlight the deployment of nodes based on a grid fashion in order to maximize sensing coverage. They propose that nodes be placed in either triangular lattices or square grids or hexagonal grids. This is also employed by Bai et al. [7] where sensors are placed in strip grids. This method of grid placement is similar to the works of Pugelli et al. [1] but is not what our work entails. Again we are not concerned with coverage but only connectivity. In their work, Wang et al. [8] use a partitioning method where there target region is partitioned into smaller regions and then nodes are deployed. Different ways of node placements are provided depending on region size. For a sensing field, sensors are deployed in the smallest number possible to maintain coverage and connectivity. For a small partitioned region, sensors are placed on the bisector of the region and for large regions, sensors are placed continuously row by row. This varies from our work as we do not partition the target space and instead of placing sensors we are placing routers. All the related work except for Pugelli et al. [1], place more emphasis on the placement of sensors. Our work assumes that sensors are placed and sensing coverage is adequate thus we are primarily concerned with the placement of routers to provide connectivity.

III. 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 [11][13] 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 preliminary solutions on the basis of algorithm. The tool then asks for the optimized algorithm and calculates synthesis process. The goal of the combination is to find candidate positions to place routers. Authors referred 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 ˆ iter do

for j=0 to 2ˆıter do

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

newNode(x,y)

for k=0 in steps of 2 to 2 ˆ iter do

for j=1 in steps of 2 to 2 ˆ 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[1].

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

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

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

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.

[1]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 GUI

draw line connecting current node to previous node

As per [1], There are a few lacunas in the mentioned algorithm

The 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 in [1].

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.

[1]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
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(

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

reduced nodes in 75dB range, node x-y coordinate)
if nextNodeFlag = 1 then
continue to next node defined by index j
if nextPathFlag = 1 then
[modified path] = buildPath(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.
[1]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.

IV. EVALUATION

Proposed synthesis algorithms are tested on 2D floor plans with multiple obstacles (walls). We did simulations 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.

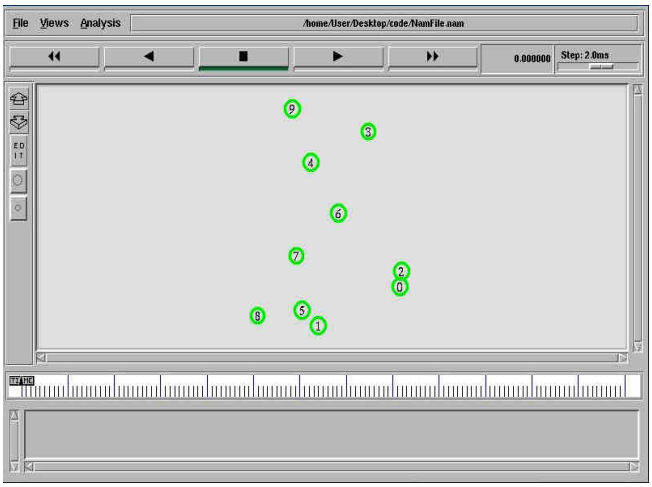

Fig 1: Final network setup based on proposed node placement

In our simulation, we assume a high dense floor plan including 10 end-devices. The older algorithm is executed with the results as percentage network not covered as 4% and routing delay is 40ms. After the implementation of the improved algorithm, 2% and routing delay is 20ms. Afterwards we try and calculate the initial placement delay for the nodes and later on, delay after shifting the nodes. We create a table as shown in table 1 for simulation results. We assume that the average distance is 10 metres.

No. of nodes	% network covered without improvement	% network covered with improvement algorithm	Delay in node placement without improvement	Delay in node placement with improvement
10	9	3	86529	5773
20	6	2	1571545	20525
30	4	2	3227487	49301
40	6	3	7824353	225601
50	6	3	13877147	1127743
60	4	1	24308082	1117937
70	8	5	44590492	1604228

Table 1: reading of simulation before and after algorithm

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

In Fig 2, we provide a graphical representation of percentage of network covered with and without the algorithm for total 7 simulations. X axis contains the total no of nodes and y axis contains overall percentage.

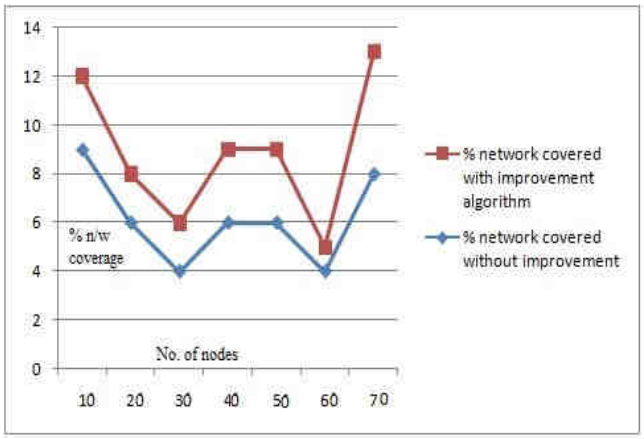

Fig.2: comparison of network coverage.

In Fig 3, we provide a graphical representation of delay in microseconds of network covered with and without the algorithm for total 7 simulations. X axis contains the total no of nodes and y axis contains overall delay.

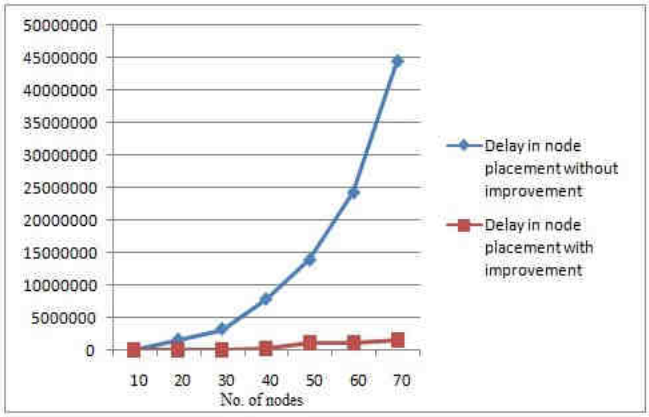

Fig. 3: comparison of delays in synthesis.

The overall improvements are presented in table II and II.

No. of nodes	% network covered without improvement(A)	% network covered with improvement algorithm(B)	Improvement in % (A-B)/B*100)
10	9	3	200
20	6	2	200
30	4	2	100
40	6	3	100
50	6	3	100
60	4	1	300
70	8	5	60

No. of nodes	Delay in node placement without improvement	Delay in node placement with improvement	Improvement in % ((A-B)/B*100)
10	86529	5773	1398.857
20	1571545	20525	7556.736
30	3227487	49301	6446.494
40	7824353	225601	3368.226
50	13877147	1127743	1130.524
60	24308082	1117937	2074.37
70	44590492	1604228	2679.561

V. Conclusion

In this Paper, we simulated the synthesis algorithm for the distributed wireless sensor networks. In Algorithm Simulations, we tried to optimize the connectivity and coverage problems. As future work, we plan to improvise the process by implementing various different algorithms as combination to validate and optimize the process 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 collected measurements on network resiliency and lifetime will allow us to further tune the synthesis strategies.

References

[1] 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.

[2] A. M. Gibney, M. Klepal, and J. T. O’Donnell. “Design of underlying network infrastructure of smart building,” in *Proc. 4th Int. Conf. on Intelligent Environments*, 2008, PP. 1-4.

[3] 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.

[4] 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 .

[5] J. Chang, P. Hsiu, and T. Kuo. “Search-oriented deployment strategies for wireless sensor networks,” in *10th IEEE Int. Symp. on Object and Component-Oriented Real- Time Distributed Computing, 2007*,(ISORC ‘07), 2007, PP. 164-171.

[6] N. Akshay, M. P. Kumar, and B. Harish. “An efficient approach for sensor deployments in wireless sensor networks,” in *Int. Conf. on Emerging Trends in Robotics and Communication Technologies*, 2010, PP. 350-355.

[7] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. H. Lai. “Deploying wireless sensors to achieve both coverage and connectivity,” in *Proc. 7th ACM Int. Symp. On Mobile ad hoc Networking and Computing*, 2006, PP. 131-142.

[8] Y. Wang, C. Hu, and Y. Tseng. “Efficient deployment algorithms for ensuring coverage and connectivity of wireless Sensor networks,” in *Int. Conf. on Emerging Trends in Robotics and Communication Technologies (INTERACT)*, 2010, PP. 114- 121.

[9] T. Clouqueur, V. Phipatanasuphom, P. Ramanathan, and K. K. Saluja. “Sensor deployment strategy for target detection,” in *Proc. 1st ACM Int. Workshop on Wireless Sensor Networks and Applications*, 2002, PP. 42-48.

[10] Elysium Ltd. “JPEG”. Internet: http://www.ipeg.org/ 2007[0ct. 2, 2013].

[11] 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

[12] 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, 2013 (Volume:PP, Issue:99), pp. 1-14.

[13] 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

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

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

[15] Y. Wang, C. Hu, Y. Tseng: Efficient Deployment Algorithms for Ensuring Coverage and Connectivity of Wireless Sensor Networks,in Proceedings. First International Conference on Wireless Internet
[16] 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 '08, pp. 515–522.
[17] 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.
[18] Pavlos Papageorgiou, "Literature Survey on Wireless Sensor Networks", pavlos@eng.umd.edu, July 16, 2003
[19] 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)
[20] 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
[21] 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.
[22] 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.
[23] 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.
[24] 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.