Sensitivity Analysis of Voronoi-based Sensor Deployment and Reconfiguration Algorithms

Gareth Nicholls, Derrick Kourie, Tinus Strauss

University of Pretoria, South Africa

Abstract

This study examines the effects of location inaccuracies on two movement-assisted Voronoi-based sensor deployment and reconfiguration algorithms, VEC and VOR, due to Wang et al. For the purposes of examining the extent to which the deployment and reconfiguration algorithms are capable of reducing coverage holes, a simulator environment was set up, using a custom-designed simulation tool. By integrating the environment with that of a GIS application, real-world distance and scaling can be applied, allowing the assessment of the algorithms to be performed in a virtual world mimicking that of a real-world deployment.
The simulation results suggest the VOR algorithm is reasonably robust if the location inaccuracies are somewhat lower than the sensing distance, and also if a high degree of inaccuracy is limited to a relatively small percentage of the nodes. The VEC algorithm is considerably less robust, but prevents nodes from drifting beyond the boundaries in the case of large inaccuracies.

CATEGORIES AND SUBJECT DESCRIPTORS:

I.6.6 [Simulation and modeling] – *Simulation Output Analysis*

GENERAL TERMS:

Algorithms, Measurement, Performance.

ADDITIONAL KEY WORDS AND PHRASES:

Geo-information system, sensors, nodes, coverage hole, Voronoi Polygon

1. INTRODUCTION

Advances in micro-electronic mechanical systems technology have seen a growth in the field of Wireless Sensor and Actuator Network (WSAN) research. These networks consist of one or more base stations and tiny nodes or motes (potentially thousands of them) that are scattered in a given region of interest (ROI), to sense and monitor the surroundings. These networks are differentiated from conventional mobile ad hoc networks not only because of their ability to sense data, but to process the data at the node level and relay the processed data to a base station[1] via the network. The nodes may also be equipped with actuators that allow them to react and perhaps change their environment, based on messages received from the base station or results from the processed data. The above sensor /actuator networks may well be deployed into harsh environments to sense and react in areas that are inaccessible to humans. The deployment of the nodes by third party systems results in random and/or unpredictable deployment (Wang *et al.* 2004, Ahmed *et al.* 2005). Such random deployment will invariably result in the occurrence of coverage holes within the ROI. Ahmed *et al.* (2005) describes the notion of a coverage hole as an area within the ROI that is not covered by at least k sensors. The degree of coverage, k, is determined by the application[2]. It is assumed that the extent of coverage in terms of internode communication signals is sufficient for all nodes to communicate within the network. It is also assumed that each node acquires information about its location (e.g. by means of GPS, triangulation or radio-location). The location of each node is then broadcast to all the neighbours within the node's communication range. Moreover, it is assumed that the position of a node can be adjusted.

Section 2 below describes the benefits of a GIS based simulator. Section 3 indicates how location information and Voronoi polygons are used to determine coverage holes in a node's vicinity. Section 4 then describes the so-called VEC and VOR-algorithms as presented by Wang *et al.* (2004). These algorithms determine how nodes should be repositioned to increase the coverage, using Voronoi polygons. The results of a simulation exercise, using a custom built GIS simulator, are presented in section 5 to indicate the algorithms' effectiveness. Section 6 focuses on inaccuracies in node location information. It shows how such inaccuracies may typically be present in a real-world context. It then uses simulation to examine the impact on the two algorithms that such inaccuracies have.

[1] Fixed components of the network containing greater computational, energy and communication abilities

[2] Within the present study coverage is assumed to be that of 1 node sensing the given area

2. GIS AIDED SIMULATION PLATFORM

For the purposes of examining the extent to which the deployment and reconfiguration algorithms are capable of reducing coverage holes, a simulator environment was set up, using a custom-designed simulation tool. A simulator application was built using the SmallWorld Architecture Framework (SWAF) and the GE Magik development language. Magik is an object-orientated language developed by GE Network Solutions as part of the SmallWorld GIS suite of applications. By developing the application using SWAF, the application becomes an extension to the existing SmallWorld Core application, allowing the simulator to interact with the maps and libraries within. Using the GIS functionality of SmallWorld Core, the simulator is able to place objects such as buildings, vegetation and nodes to an accuracy of 1mm within the GIS environment. The GIS introduces realistic distances and locations within a given ROI. Given this, the simulator can deploy *n* nodes in a real-world layout, monitoring the robustness of the coverage protocols taking into account location inaccuracies, distance moved, possible signal /coverage strength and energy consumption. The following figure shows a screen shot of the SmallWorld Core application (a) and the custom built GIS simulation application (b).

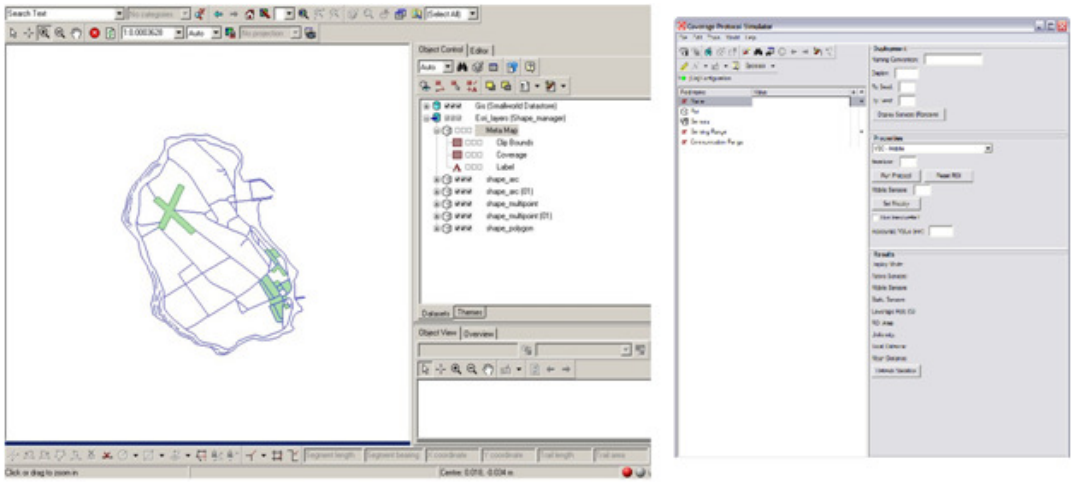

Figure. 1. ***SW Core application and the custom built simulation application.***

3. VORONOI POLYGONS

The location information about proximity to neighbours allows for the construction of a (unique) Voronoi polygon around each node (Aurenhammer, 1991). This section discusses how the Voronoi polygons are used to determine the existence the coverage holes within the ROI.

A Voronoi polygon of a node has the property that each point in it is closer to its associated node than to any other node in its surroundings. A Voronoi-diagram, a decomposition of the ROI, is the result of determining all the Voronoi polygons in the ROI (Aurenhammer, 1991). By applying the properties of the Voronoi polygons it can be said that each Voronoi polygon indicates a local area of coverage for which a node should be made responsible. The area of such a Voronoi polygon that falls outside the circle of coverage of its associated node can be used to determine the overall size of the coverage holes in the network - Figure. 2.

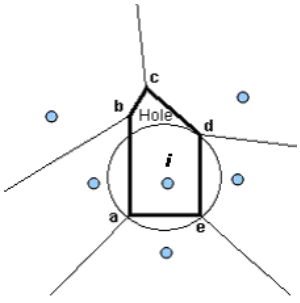

Figure. 2. ***Determining the existence of a coverage hole.***

The total coverage hole can be determined by equation [1]. It shows that the total coverage hole, H, is equal to the area of the ROI less the union over all sensors 1 to N, of their representative coverage areas, Si. Note that for each i, Si excludes any coverage that lies outside the ROI.

$$H = ROI^2 - \bigcup_{i=1}^{n} Si \qquad [1]$$

4. MOVEMENT-ASSISTED SENSOR DEPLOYMENT PROTOCOLS

By Voronoi-based deployment and reconfiguration algorithms, we refer to algorithms that construct Voronoi polygons to determine the existence and extent of coverage holes, and then calculate the movement of the nodes based on the Voronoi polygon with the aim of

minimizing holes. We refer to research such as Khan *et al.* (1999), Heo *et al.* (2003) and Wang *et al.* (2003) for active movement-assisted networks. For the purpose of this study we examine the movement-assisted sensor deployment and reconfiguration algorithms presented by Wang *et al.* (2004).

4.1 The VECtor-based Algorithm (VEC)

The VEC algorithm is a *push-based* algorithm in that it pushes the neighbouring nodes away from each other. The algorithm is inspired by the behaviour of electro-magnetic particles: when two particles are too close to each other they exert a force pushing them apart (Wang *et al.* 2004).

Nodes are assumed to be optimally placed when they are evenly distributed within the ROI, each one being at some constant distance, d_{avg}, from its neighbours. Since the number of nodes and ROI size is known, this value may be precomputed. Suppose that $d(S_i,S_j)$ is the distance between sensor S_i and S_j. If $d(S_i,S_j) > d_{avg}$ and if S_j is within communication distance of S_i, then VEC assumes that S_i and S_j mutually exert a "virtual force" on one another that is proportional to $(d_{avg} - d(S_i,S_j)) / 2$. In general, this virtual force, cumulatively determined for node S_i, determines the distance and direction that S_i moves in each iteration of the algorithm.

However, there are a number of special considerations. In the first place, if S_i already covers its local area as defined by the Voronoi polygon, then it will not be move. Instead, the force $(d_{avg} - d(S_i, S_j))$ will be exerted on the node S_j only. Secondly, to prevent the nodes from moving too close to the boundary, an additional force is generated by the boundary of the ROI. The boundary force will push the node to move $(d_{avg} / 2 - db(S_i))$, where $db(S_i)$ is the distance of S_i to the ROI boundary - Figure. 3.

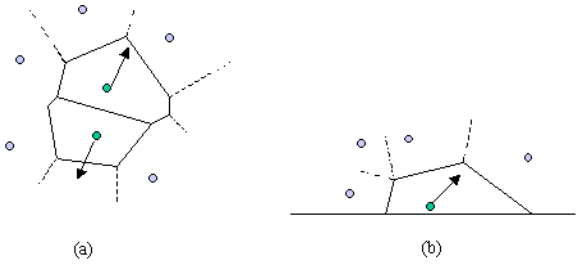

Figure. 3. ***(a) Virtual Forces between two sensors, (b) Virtual Force exerted by a boundary.***

To prevent greater coverage holes from forming due to the movement of a node, Wang *et al.* (2004) introduce *movement-adjustment.* After the final virtual force on a sensor has been determined, the local coverage is recalculated based on the potential movement. If the coverage is not improved, a midway point between the node's current location and calculated location is examined. If the local coverage is increased at this new target location then the node is moved accordingly; otherwise the node remains in the current position for one iteration. A further check is put in place to prevent the node from moving outside the ROI. If the node location is placed outside the ROI, then the node remains in the current position for one iteration.

The VEC algorithm thus runs iteratively until a given threshold is reached. The nature of the threshold is explained below. Each iteration consists of two phases - a discovery phase and a movement phase. During the discovery phase each node broadcasts its location to its neighbours and also calculates its polygon from the information received from its neighbours. After this phase the node's position is adjusted in the movement phase.

4.2 The VORonoi-based Algorithm (VOR)

The VOR algorithm is a *pulled-based* algorithm in that it pulls the sensors to their local maximum coverage hole. Once the node detects the existence of a coverage hole, the node then moves towards the farthest vertex of the relevant Voronoi polygon. The distance that the sensor moves, denoted as V_{far}, is calculated as the distance to the farthest vertex, less the sensing radius of the node, For example, in Figure. 4, $V_{far} = d(S_i,A) - rS_i = B$, where $d(S_i, A)$ is the distance from node S_i to point to A, and rS_i, is the sensing radius of the node. In the figure, B, denotes the point to which the node should move. The distance moved is limited to be at most half of the communication distance (as opposed to the sensing distance). This avoids situations where the node's local view of the Voronoi polygon does not know of the existence of a neighbour, due to communication limitations, thus moving too close to the neighbour's sensing area (Wang *et al.* 2004).

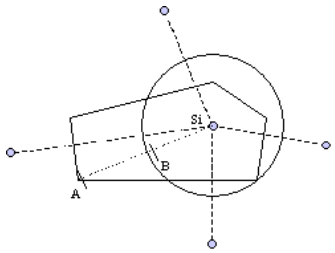

Figure. 4. ***Movement of nodes using VOR (Wang 2004).***

Both VEC and VOR can be classified as a greedy algorithm in that they attempt to reduce the largest holes. However, the movement of a node to solve a hole in one direction could potentially cause another hole in the opposite direction, resulting in *movement oscillations*, so that the sensor moves back and forth continuously between two points. To prevent this, the algorithms introduce *oscillation control*. The previous direction of movement of each node is stored. Each

time a node is to relocated, a check is first made to verify that the next move is not in the opposite direction to the previous move made. If this holds true, then the node remains in the current position for one iteration.

The VOR algorithm shares the same runtime attributes as VEC in the sense that it runs iteratively until a given threshold is reached. Each iteration consists of the same two phases as VEC, a discovery phase and a movement phase.

5. CONTROL EXPERIMENTS

The experiments described in this section are similar (but not identical) to those described in Wang *et al.* (2004). This experiment is based on conditions, which are deliberately chosen as "ideal". The experiment is therefore used to provide a control or base-line scenario in considering the performance of both algorithms in relation to later experiments that are discussed in section 6. An environment is simulated in which 25 nodes are deployed randomly within an ROI that has an area $1615m^2$. Each node was assigned a sensing range of 5 meters and double that in communication range – i.e. 10 meters. The nodes were deemed to be capable of self-movement, and to be unaware of power consumption. It was also assumed that all the nodes were deployed without failure, and that sensing and communication took place within the respective radii without interference.

These initial assumptions raise the question: how robust are the algorithms with respect to randomly chosen starting positions, as well as with respect to long term iterative behaviour?

5.1 Terminating Conditions

In the initial control experiments, a limit on the mean total distance travelled by the nodes served as a termination condition for the iterations. This mean distance is found after each iteration, by summing the distance travelled by each node from its *initial* to its present position, and then dividing by the number of nodes. If this mean distance travelled is greater than the limit, then the algorithm is terminated; otherwise another iteration is executed. A mean total distance of 10 meters was chosen as a terminating condition for both algorithms. This corresponds to the communication range that had been assigned to nodes. Under this terminating condition, both algorithms executed four iterations.

The results for VEC were as follows. The original coverage hole was calculated as 31.57%. After the fourth iteration the coverage hole had been reduced to 16.21% of the ROI. This means that the coverage hole was reduced by about 49% of its original size. The experiment was repeated using VOR. The same starting positions for the nodes were used. After the fourth iteration the coverage hole had been reduced to 14.12% of the ROI. In this case, the coverage hole reduction was slightly larger – about 55% of its original size.

Thus, both algorithms achieve considerable improvement from the starting scenario, with VOR doing somewhat better than VEC. These results compare with the results obtained by Wang *et al.* (2004). The next subsection tests the robustness of each algorithm with respect to the initial deployment of nodes within the ROI.

5.2 Random Initial Deployment

Each algorithm's behaviour is tested from each of fifteen randomly determined initial deployment locations. Coverage holes ranged between 32.5% to 44.7%. As oppose to using the 10-meter terminating condition of section 5.1, each algorithm was run for 100 iterations for each of the 15 starting scenarios. This means that nodes were allowed to drift as far from their original position as was dictated by 100 iterations of the respective algorithms.

The results for the VEC algorithm are shown in Figure. 5. The graph displays – for the 15 starting scenarios – the average, maximum and minimum coverage hole size at each iteration. All initial scenarios generally improve over the 100 iterations, the final average, maximum and minimum coverage hole percentages being 17.53%, 21.68% and 14.4% respectively. Although improvement is not guaranteed from one iteration to the next, after approximately 20 iterations the average improvement stabilises to approximately 20%. No starting scenario causes the VEC algorithm to diverge as the number of iterations increased. At about 90 iterations, the minimum coverage hole of about 10% is attained.

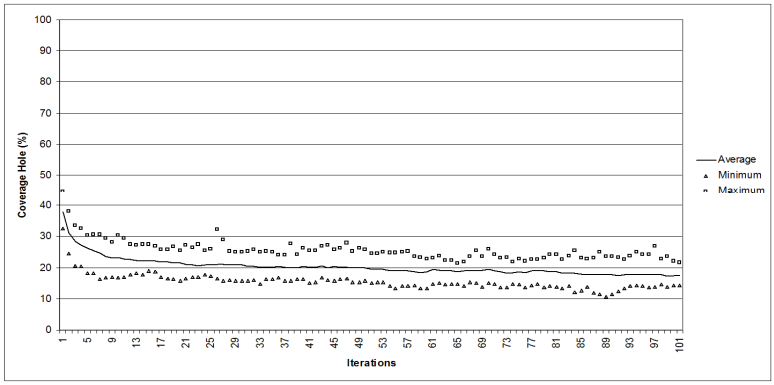

Figure. 5. ***Randomly deployed starting scenarios for the VEC algorithm.***

The results of an identical experiment based on the VOR algorithm are shown in Figure. 6. The overall picture is similar to that provided by the VEC algorithm, although convergence (below 10% coverage hole after 15 iterations) and overall performance is somewhat better. The final average, maximum and minimum coverage hole percentage is 8.3%, 12.6% and 4.15% respectively, and a minimum coverage of less than 5% is attained several times after 60 iterations.

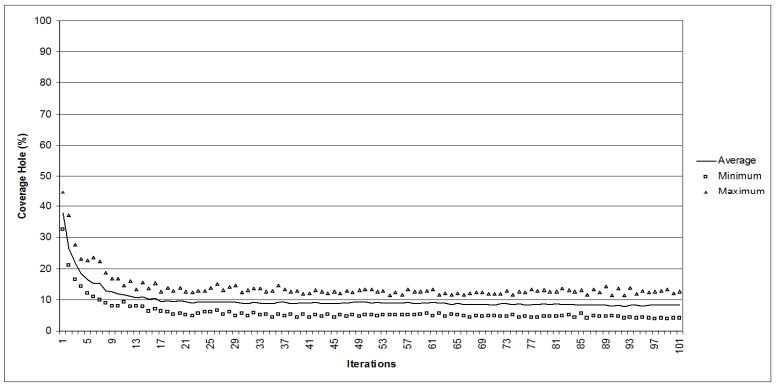

Figure. 6. ***Randomly deployed starting scenarios for the VOR algorithm.***

These results are broadly commensurate with results in section 5.1, which relied on a different initial starting scenario and termination condition: the 14.12% coverage hole attained by VOR after four iterations using the 10-meter terminating condition is within the range attained by the other fifteen starting scenarios after 100 iterations; while the 16.21% attained by VEC falls slightly outside the maximum (21.68%) attained by the other fifteen starting scenarios after 100 iterations, both algorithms remained close to the average coverage hole percentage calculated over the fifteen starting scenarios. Most significantly, however, in no case was any divergence observed over long-term iterations.

This provided us with a measure of confidence that further sensitivity analysis on the control scenario could provisionally be regarded as representative of general VEC and VOR behaviour.

In the control experiment the performance of the algorithm was observed for the case where the nodes have accurate location information. In the forthcoming section location inaccuracies are introduced, and performance is contrasted with this present control.

6. LOCATION SENSITIVITY

An optimal approach to determine accurately the location of a node would be to use *geo-location* by fitting a *Global Positioning System* (GPS) to each node in the network. However this solution adds to the manufacturing cost of each node as well as to the energy consumption of the node, and is thus generally not considered practical. Thus, normally in a real-life situation, some location algorithm is used to determine the location of nodes within the network. Triangulation and radio-location are two possibilities. However, in applying these algorithms, inaccuracies in the calculation of location are likely to occur (Nicules and Nath, 2003).

The current study is only concerned with the degree to which inaccuracy could occur within the localisation protocols. For this reason a simplistic method to simulate inaccuracy is used. Node S_i at location l_{Si} was injected with a location inaccuracy by choosing a random position on the perimeter of a circle of radius x with centre at l_{Si}. This is illustrated in Figure. 7 where r_{Si} indicates the new and inaccurate location of l_{Si}. Note that these inaccuracies were injected into the positioning of a node as part of the discovery (first) phase of each iteration of each algorithm. In this way, incorrect location data broadcast by nodes is simulated. Clearly, the calculation of the Voronoi polygons and movement location will be effected by the inaccuracy.

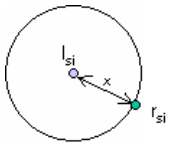

Figure. 7. ***Simulating inaccuracy calculation.***

In the present research, the GIS based simulator was used to run the VEC and VOR algorithms under different degrees of location inaccuracy. The intention was to determine the effect of inaccuracy on each of the algorithms.

6.1 Graduated Inaccuracies

The simulations of Section 5 assumed a best-case scenario – one in which all the nodes in the network accurately reported their position to the same extent. The same initial deployment setup as before was used to run each algorithm with various levels of inaccuracy. Twenty simulations were run per algorithm. In each case, gradual increases in the number of inaccurate nodes were set. Figure. 8 and Figure. 9 show the effect on the coverage holes as the number of nodes (of the original 25) that report inaccurately increases from 0 to 25.

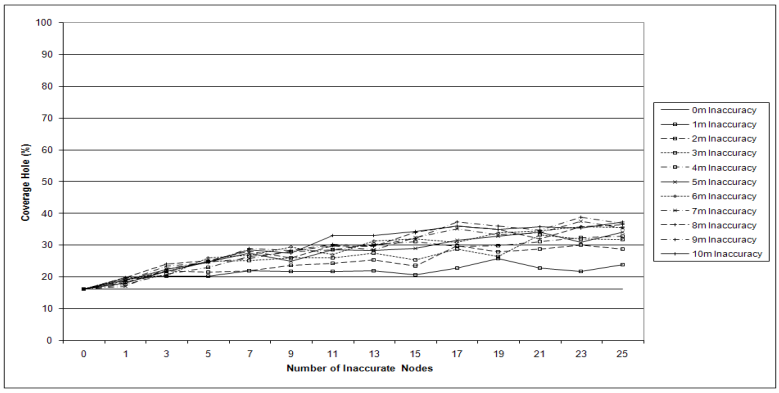

Figure. 8. ***Increasing inaccurate nodes (VEC).***

Figure. 8 shows the VEC algorithm's degree of recovery as the number of inaccurate nodes increases. When there are no accurate nodes, the coverage hole varies between about 23% and 37%, depending on the level of inaccuracy. When 20 out of the 25 nodes report accurately, at 2 meters inaccuracy the coverage hole drops to around 20%, while at 4, 6 and 8 meters inaccuracy the final recovery is the same at about 26%. The slopes of graphs generally indicate a gradual increase in coverage hole per new inaccurate node introduced. Very broadly, one could say that each inaccurate node decreases the VEC algorithm's effectiveness by about 1%.

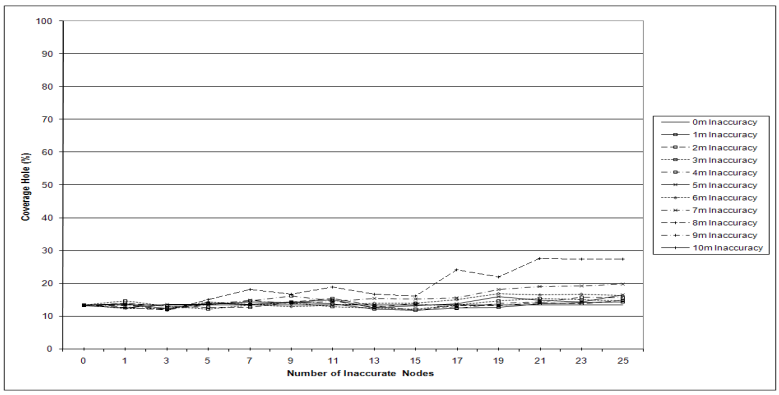

Figure. 9. ***Increasing inaccurate nodes (VOR).***

As can be seen in Figure. 9, the VOR algorithm generally performs better in the presence of inaccuracies than the VEC algorithm. The VOR algorithm seems reasonably tolerant of inaccuracies, even fairly large ones, provided that number of inaccurate nodes is limited. By fairly large is meant inaccuracies (6 to 8 meters) that are of the same order of magnitude as the sensing radius (5 meters). When 5 nodes (i.e. 20% of the 25 nodes) were inaccurate at a level of 8 meters, the coverage hole was about 20% of the ROI – a mere 6% degeneration from the control coverage hole of 14.12%. On the other hand, at 2 and 4 meters inaccuracy, the algorithm's performance appears to be reasonably indifferent to the number of accurate nodes, suggesting that VOR is quite robust in the presence of relatively small inaccuracies. In these cases, the coverage hole remains close to the control of 14.12%, and indeed, in certain instances drops below it.

6.2 Worst Case Inaccuracy

The following simulation attempts to assess the algorithms in an extreme worst-case scenario. In each case, *all the nodes* were set at an inaccuracy level of *n* meters, where the inaccuracy ranged from 1 to 10, increasing in 0.5 meter intervals. The termination criterion used in the control experiment was retained. For each inaccuracy level, say of *n*, the algorithm was run 50 times. Each such run located each node in each iteration at a displacement of *n* meters away from the node's previous position. The minimum, maximum and average coverage holes were computed over these 50 runs.

Figure. 10 shows the results obtained for the VEC algorithm. The coverage hole rapidly increases as the inaccuracy level is increased – i.e. inaccuracy has a pronounced effect on coverage. For example, at an inaccuracy of 2.5 m the initial accurately-determined coverage hole of 16.20% more than doubles. As inaccuracies increase to 5 meters per node, the coverage hole grows to about 35%.

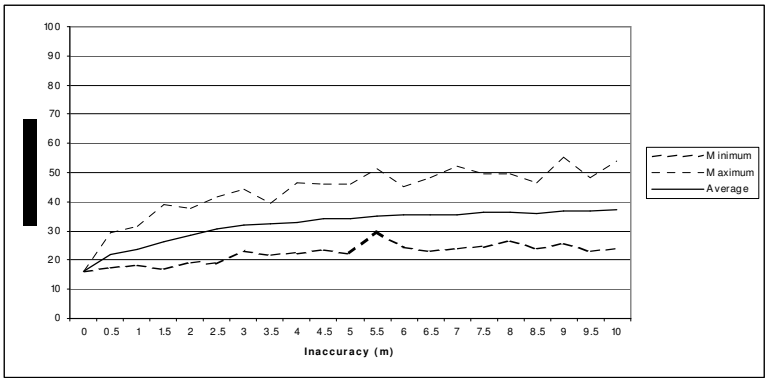

Figure. 10. ***Coverage hole(%) related to an increase in location inaccuracy (VEC).***

Interestingly, the coverage hole remains at around 37% for higher inaccuracies. This is evidently due to the way in which the VEC algorithm deals with the ROI boundary. If the node determines its new location to be outside the boundary, then the VEC algorithm does not allow the node to move.

Figure. 11 shows the results obtained for the VOR algorithm. In this case, the coverage hole percentage recovery is relatively robust for inaccuracies up to about 6 meters. Thereafter, the inaccuracy has a relatively pronounced effect on coverage. As the location inaccuracy increases, so does the coverage hole. A coverage hole of about 16% for an inaccuracy level of 6 meters, increases to more then 26% when the inaccuracy doubles to 8 meters. This is clearly due to the VOR algorithm's dependency on location both during the discovery phase when creating the polygons, as well as during the movement phase in determining the position to which the node should move.

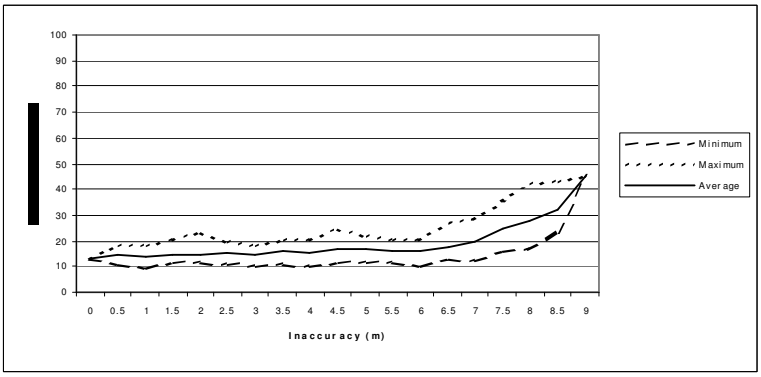

Figure. 11. ***Coverage hole (%) related to an increase in location inaccuracy (VOR).***

6.3 Inaccuracy over Long-term Iteration

The foregoing described experiments that were based on a termination criterion applied to both the VEC and VOR algorithms, whereby the total average node movement relative to original node position was limited to 10 meters. It seemed important to verify that this did not represent some artificial termination point, and that significant coverage improvement could not perhaps be gained, even in the face of inaccuracies, by increasing the number of iterations. In Figure. 12 and Figure. 13, the coverage is shown under the various worst-case inaccuracy scenarios as the number of iterations was increased to 100.

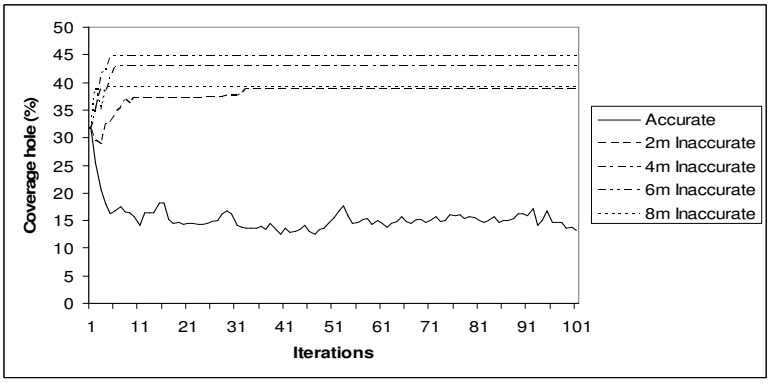

Figure. 12. ***Coverage hole (%) by number of iterations (VEC).***

Both in Figure. 12 and in Figure. 13, it is evident that, under none of the scenarios, are significant gains to be had by increasing the number of iterations. For example, in Figure. 12 (respectively Figure. 13), the simulation cases for the accurate locations shows a 16.20% (14.12%) coverage hole after four iterations and 13.22% (11.86%) after 100. In the presence of any form of inaccuracy, the VEC algorithm diverges, and all nodes eventually drift to the boundary.

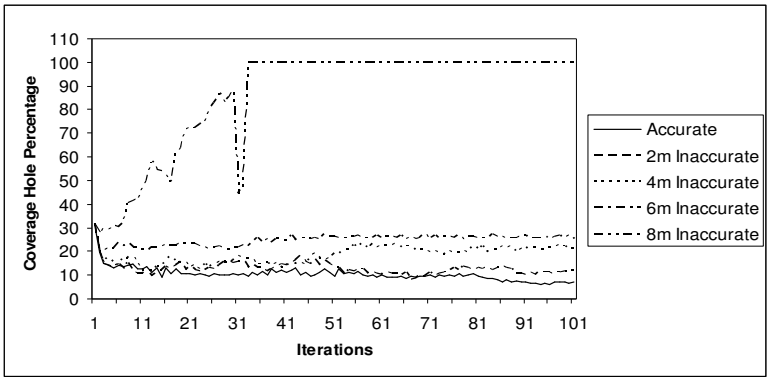

Figure. 13. *Coverage hole (%) by number of iterations (VOR).*

The VOR algorithm is somewhat more stable except in the presence of high inaccuracy (8 meters), in which case the algorithm diverges, and eventually all nodes drift out of the ROI.

7. CONCLUSION

In this study, the focus was on the assessment of the VEC and VOR algorithms' sensitivity to inaccurate location information. Our results suggest that the VOR algorithm is reasonably robust if the inaccuracies are somewhat lower than the sensing distance. It remains reasonably robust when the inaccuracies are somewhat higher, provided that they do not affect a very high proportion of nodes. On the other hand, the VEC algorithm shows a high dependency on the accurate calculation of node locations. However the algorithm acquires a certain type of robustness in relation to node behaviour at the ROI boundary. By keeping all the nodes within the ROI, in the worst case, nodes may cluster around the boundary, but a coverage hole of 100% can never occur.

These results are useful in two ways. Firstly, they suggest that the algorithms (the VOR algorithm, in particular) can be applied with some degree of confidence to better position a network of movement-assisted nodes, even if the location information is not completely reliable. Secondly, the results suggest an approach to node placement in the first place: simulate the optimal positioning of nodes as has been done in this exercise. Then attempt to place the nodes in locations suggested by the simulation, but with a degree of confidence that moderately erroneous placement is unlikely to have a great impact on the extent of coverage.

Current work includes the assessment of a further three algorithms from static and hybrid network deployments. These algorithms include the Coverage-Preserving Node Scheduling Scheme (CPNSS) presented by Tian et al. 2002, the Optimal Geographical Density Control algorithm by Zhang and Hou (2003) and the Bidding Protocol developed by Wang et al. 2003. Much work has been done using Voronoi diagrams to equally distribute points within a Euclidean space, such as the principles applied in Lloyd's algorithm, also known as Voronoi iterations. An interesting research direction would be to compare convergence rates between the algorithms presented here as well as problems solved by Lloyd's algorithm.

Work is in progress to extend the simulator to take into account real-world objects such as buildings and landscape. The effects of the real-world objects in terms of signal strength and radio interference will also be taken into account. The sensor node objects will also be enhanced to consider energy consumption. Additionally, simulations based on various other Voronoi based algorithms are under development. Also under consideration is simulated location sensi-tivity analysis in respect of other placement algorithms that do not rely on the construction of Voronoi polygons.

REFERENCES

1. Ahmed, N., Kanhere, S. S. and Jha, S.(2005): The Holes Problem in Wireless Sensor Networks: A Survey. *Mobile Computing and Communication Review*, vol 9 no 2 pp 4-18.
2. Akyildiz, I.F. and Kasimoglu, I.H. (2004): Wireless sensor and actor networks: research challenges. *Georgia Institute of Technology*, Georgia, USA.
3. I.F. Alkyildiz, W. Su and Y. Sankarasubramaniam, (2002): Wireless Sensor Networks: A Survey, *Computer Networks*, vol 38, issue 4, pg 393-422
4. J. Albowicz, A. Chen and L. Zhang, (2201): Recursive position estimation in sensor networks, *The 9th International Conference on Network Protocols*, pg 35-41
5. Aurenhammer, F. (1991): Voronoi Diagram – A Survey of a Fundamental Geometric Data Structure. *ACM Computing Survey*, 23, ACM Press.
6. de Silva, V., and Ghrist, R. (2007): Homological Sensor Networks. *Notices of the American Mathematical Society* 54
7. GE SmallWorld Core Documentation Version 4, GE Power Systems, 2003
8. D. Harel, "Algorithmics – The Spirit of Computing", 2nd edition, Addison-Wesley Publishing Company, 1996
9. N. Heo and P.K. Varshney, "An Intelligent Deployment and Clustering Algorithm for a Distributed Mobile Sensor Network". In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics,* volume 5, pages 4576-4581, October 2003.
10. M. Kennedy, "The Global Positioning System and GIS: An Introduction", *Ann Arbor Press, Inc,* Michigan USA, 1996, ISBN 1-57504-017-4
11. J.M. Khan, R.H. Katz and K.S. Pister, (1999): Next Century Challenges: Mobile Networking for Smart Dust, *In Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pg 271-278
12. (a) S. Meguerdichian, F. Koushanfar, M. Potkonjak and M.B. Srivastava (2001): Coverage problems in wireless ad-hoc sensor networks, *INFOCOM 2001, 20th annual joint conference of the IEEE Computer and Communications Societies, Proceedings IEEE*, Vol 3, pg 1380 – 1387

13. Nicules, D. and Nath, B. (2003): Ad-hoc position system (APS) using AoA. *In proceedings of the IEEE INFOCOM.*

14. P. Santi (2005): Topology Control in Wireless Ad Hoc and Sensor Networks *ACM Computing Surveys, Vol 37, No 2 '05*, pp. 164-194.

15. D. Tian and N.D. Georganas, (2002): A Coverage Preserving Node Scheduling Scheme for Large Wireless Sensor Networks *In Proceedings of the first ACM WSNA* '02, September 2002.

16. Wang, G., Cao, G. and La Porta, T. (2004): Movement-assisted Sensor Deployment. *In proceedings of the IEEE INFOCOM.*

17. G. Wang, G. Cao and T.L. Porta (2003): A Bidding Protocol for Deploying Mobile Sensors *In 11th IEEE International Conference on Network Protocol* ICNP'03, pages 315-324

18. Neo, N. and Varshney, P.K. (2005): Energy-Efficient Deployment of Intelligent Mobile Sensor Networks. *IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans*, Washington DC, USA, 35.