An Efficient Self-Organizing Node Deployment Algorithm for Mobile Sensor Networks

Mahsa Sadeghi Ghahroudi, Alireza Shahrabi, and Tuleen Boutaleb

School of Computing, Engineering and Built Environment
Glasgow Caledonian University, Glasgow, UK
Emails:{ Mahsa.Sadeghi, A.Shahrabi, T.Boutaleb}@gcu.ac.uk

***Abstract*—Wireless Sensor Networks (WSNs) constitute the platform for a broad range of applications, such as those related to national security, surveillance, military, health care, and environmental monitoring. Maximising coverage using resource-constrained nodes is usually a goal to provide the expected quality of service for these applications. This problem has been studied extensively in recent years, especially when the connectivity and energy efficiency are of high significance. In this paper, we propose a new distributed move-assisted algorithm, called SODA, to efficiently provide the maximum coverage for WSNs with self-organising mobile nodes. SODA is based on a deployment algorithm recently reported in the literature which is inspired by the equilibrium of molecules. However, while SODA's transition from chaos to order is faster, the final coverage provided by SODA is also insensitive to the initial deployment of the nodes and no specific level of coverage during the initial deployment is required. This is achieved by detecting the local network density and adjusting the partial force applied at each step in each neighbourhood accordingly. Our extensive simulation study shows the advantages of SODA including lower power consumption, as well as faster and more effective coverage.**

Keywords–coverage; distributed wireless sensor network; energy efficiency; node deployment.

I. INTRODUCTION

Wireless Sensor Networks (WSNs) are being used in many different applications in the world, particularly with the proliferation of Micro-Electro-Mechanical Systems (MEMS) technology which has promoted the development of smart sensors [1]. These applications vary from entertainment, travel, retail, and industry to medicine, and emergency management [2] [3].

In WSNs, providing adequate coverage is a fundamental problem that has gained much attention recently [4] with the aim to provide the maximum sensing coverage over the Region of Interest (ROI). The required coverage can be achieved by proper deployment of sensors after the initial deployment. Therefore, random deployment of mobile sensors does not guarantee the expected coverage in the ROI.

The mobile sensor deployment algorithms in WSNs are classified as centralised or distributed [5]. The distributed method is fault tolerant, scalable, and cost-efficient and, hence a more popular method in wireless sensor networks [6] [7]. Distributed deployment of sensors does not rely on a centralised node, i.e., sink, to decide on all the sensors movements. In the distributed strategy, every sensor can communicate with its neighbouring nodes to decide about its movement at each step. The communication method, the data sent and received, and also the communication and sensing range are some of the essential parameters to be specified in every distributed deployment algorithm.

In different algorithms, different methods are used for communication between neighbouring nodes to increase the coverage in the area [7]–[12]. Some algorithms are inspired by observing some natural phenomenon behaviours to cope with the distributed sensor network requirements. For example, neighbourhood movement theory that is seen in the animal aggregation movements, like birds migration, is applied in a deployment algorithm proposed in [13]. In this deployment algorithm [13], sensors move based on the average of the neighbour's positions and, as a result, create a uniform sensor placement to achieve the required coverage. In another study [14], an algorithm is proposed for a distributed sensor network in which the equilibrium of molecules inspires sensor movement. This algorithm can provide full coverage after a rather high number of steps and provides a somewhat high initial coverage percentage over the initial deployment. Another deployment algorithm using a clustering approach to achieve better power usage and coverage has also been proposed in [14] with the same concept for sensor movements. However, none of them considers assumptions like the initial deployment of sensors in the antagonistic environment in which manual deployment of sensors is not possible. For instance, a sensor network could be deployed near the crater of a volcano to measure temperature, pressure, and seismic activities [15] or it could be deployed as part of security surveillance in military operations [6]. Therefore, the initial deployment of sensor nodes with a certain percentage of initial coverage is not always possible.

In this paper, we aim to develop a distributed deployment algorithm, called Self-Organizing Deployment Algorithm (SODA) for mobile sensor networks. Our primary goal is to achieve maximum coverage within an acceptable range of energy consumption and time cost. The SODA is based on the Distributed Self-Spreading Algorithm (DSSA) [14], which is inspired by the equilibrium of molecules. SODA addresses the DSSA limitations, such as sensitivity to the minimum percentage of initial deployment coverage, the long transition time from chaos situation to order, and limitations in providing the appropriate coverage for single-point-deployment scenarios, where all nodes are initially located over a single sub-area, such as one corner or at the centre of the area. The superiority of SODA is due to the adjustment of partial force where the network is dense locally.

The rest of the paper is organised as follows. Section II presents an introduction to DSSA followed by the SODA

 ISBN: 978-1-61208-676-7

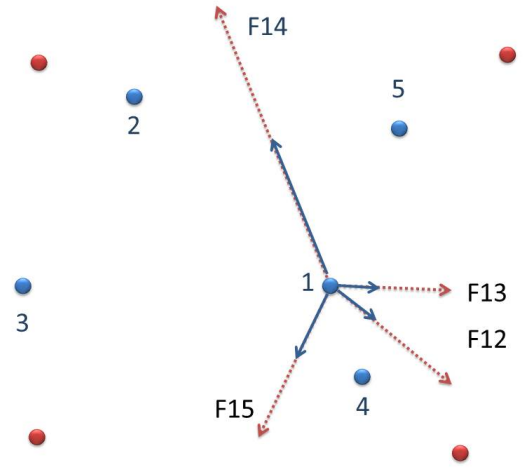

Figure 1. Partial forces in between sensor 1 and its neighbours.

solution in Section III. The performance evaluation section to describe the simulation specification and results is under Section IV while our conclusions are drawn in Section V.

II. A Brief Reviewing of DSSA

The DSSA is inspired by the equilibrium of molecules, which balances the energy of the particles to remain in their locations. This balance is caused by particles, which stay in their lowest energy point with the same distance from each other in a distributed manner. Similarly, the optimal spacing in between sensors creates the required coverage.

In DSSA, it is assumed that sensors are randomly distributed in an area. Sensors can communicate with each other if they are located within their communication range, called C_R, and can sense their environment with the sensing range, S_R. The communication between sensors is performed in order to find their neighbouring nodes and to exchange the requisite information. The collected information is then used to decide about the appropriate location of each node to provide the required coverage for the area. Every sensor node executes the DSSA after an initial step.

In the initial step of the DSSA, the communication range, C_R and sensing range, S_R values are given, which are dependent on sensors' specifications that are the same for all the sensors. The initial location of sensors is specified in a 2D vector (p_0). The higher dimension for sensor locations, is possible by adding another component to the location vector. The $threshold1$ and $threshold2$ are two parameters that can check the exit points of the algorithm and should be defined in the initial step, which are explained in the following sections. Another important variable is $\mathcal{M}$ which is the expected density in the sensor network. Expected density is the average number of sensors in a one-hop neighbourhood. The expected density is calculated by $\mathcal{M} = (\frac{N.\pi.C_R^2}{A})$, where C_R is communication range, S_R is sensing range, N is the number of sensors, and A is the size of the ROI. In addition to expected density, D represents the local one-hop neighbourhood density of every sensor. The expected density and local density control the movement of the sensors.

The central core of DSSA is the partial force that moves the sensors. This force is dependent on the current location of sensors, the distance in between every two neighbouring sensors, and the local density. Local density and partial force have a direct relationship, that is the same as the movement of particles in physics, which follow Coulomb's Law.

The partial force at step n for a sensor and its neighbouring sensor is a repulsive force calculated as:

$$f_(i,j) = \frac{D^i}{\mathcal{M}^2}(C_R - |p_n^i - p_n^j|)\frac{p_n^j - p_n^i}{|p_n^j - p_n^i|} \quad (1)$$

p_n^i stands for the position of sensor i at step of n, and D^i stands for the local density of sensor i at step of n. As appears in (1), the magnitude of partial force depends on the position of the nodes. If any neighbour of the sensor, like sensor j, has a higher value in one dimension then the magnitude of that force is positive and negative otherwise.

For every sensor node, the total force that moves that sensor is the cumulative force of all neighbouring nodes in its one-hop neighbourhood. The movement is independent of the movement of any other sensor node inside or outside of the neighbourhood. This process is executed as long as the conditions for stopping the algorithm are not satisfied. These conditions are:

- *Oscillation Check*: The execution of the algorithm at each node is stopped when it reaches its oscillation limit. Oscillation happens when a sensor moves back and forth between almost the same locations consecutively. The number of oscillations is counted by oscillation count, O_{count}. The distance that a sensor moves back and forth is called $threshold1$, and oscillation limit, O_{lim}, is the maximum number of oscillations until the sensor stops its movement. A sensor stops its movement if its O_{count} equals the O_{lim}.
- *Stability Check*: If a sensor moves less than a $threshold2$ over a number of steps, it can be concluded that it has reached its stable position and it can stop its movement. To count the number of these steps, a variable, $StabilityLimit(S_{lim})$, is defined. The stability check is useful if a sensor breaks down or has reached stable status.

III. Self-Organizing Deployment Algorithm (SODA)

A. DSSA limitations

The concept of partial force and the background idea of equilibrium of molecules, make DSSA a prominent solution in WSN to achieve full coverage. However, many assumptions in DSSA are not feasible. Sensitivity to the minimum percentage of initial coverage, initial uniformity, non-single-point-deployment are some assumptions in DSSA, which cannot be applicable in the realistic scenarios. In addition to these assumptions, the high order of time for DSSA from chaos to order state has been observed as one of the challenges. These assumptions and challenges in DSSA are:

1) Single-point-deployment: In many applications in sensor networks like chemical sensitive environment or borders, single-point-deployment is the only way to initially locate the sensors since it is not feasible to uniformly locate the sensors. In DSSA, the initial deployment of sensors is considered to be uniformly distributed in the ROI where in some

 ISBN: 978-1-61208-676-7

```
1: procedure INITIALIZE
2:     P ← an array includes all the sensors locations
3:     C_R ← communication range of sensors
4:     S_R ← sensing range of sensors
5:     D ← calculated local density
6:     M ← calculated expected density
7: end procedure

8: procedure SODA(Until all the sensor nodes are stable)
9:     Calculate local density for the sensor
10:    if Sensor i is not stable then
11:        if Local density > Expected Density (M) then
12:            Partial Force is ← Fnew_n(i, j)
13:        else
14:            Partial Force is ← f_n(i, j)
15:        end if

16:        Update next step position for the sensor

17:        Check for oscillation
18:        if Oscillation happens then
19:            Increase O_count
20:            if O_count > O_limit then
21:                Move the sensor to centriod of
                   oscillation points and make it stable
22:            end if
23:        end if

24:        Check for stability
25:        if Sensor node is stable then
26:            Increase S_count
27:            if S_count > S_limit then
28:                make the sensor stable
29:            end if
30:        end if
31:    end if
32: end procedure
```

Figure 2. SODA Algorithm.

environments random deployment of sensors is not possible. Therefore, single-point-deployment is a necessity in most of the applications, which initially causes a high local density in a part of ROI.

The partial force in DSSA depends on the local density and distance in between sensors. An illustration of a sensor node and its neighbours in the DSSA is shown in Figure 1. The blue forces are for a scenario where $sensor1$ and 4 blue sensors in the $sensor1$'s neighbourhood are in the area, and the red forces are for another scenario that 4 more red sensors are added in the current area. The size of the partial forces has a direct relationship with their distance to the $sensor1$ and the local density. Therefore, the partial force for a closer sensor like sensor 4, $F14$, is greater than, the partial force for sensor 2, $F12$. In another scenario, if the local density of a sensor increases by adding some sensors (i.e. red sensors in Figure 1), the values of the partial forces even for previous sensors increase, that are shown as red forces.

For scenarios where the density is quite high in a sub-region, the high force moves all the sensors with an unreasonable intensity to the corners. The $\frac{D}{\mathcal{M}^2}$ shows the density factor in the partial force. The $\frac{D}{\mathcal{M}^2}$ parameter is large when sensor i is surrounded with many sensor neighbours. In the dense areas $(C_R - |p_n^i - p_n^j|)$ affects the partial force inversely. The adjacent sensor node creates a larger force in comparison to the sensor, which is far apart and not in the dense area. Therefore, the intense move that is caused by $\frac{D}{\mathcal{M}^2}$ is known as the density factor, which is large in this case. Also the small distance between sensors, which causes $(C_R - |p_n^i - p_n^j|)$ parameter to be closer to C_R. Addressing these issues can improve the performance of the deployment algorithm for all scenarios including the single-point-deployment cases.

2) Sensitivity to the minimum percentage of initial coverage: In order to achieve the best coverage, a minimum coverage during the initial deployment is required in DSSA. The random deployment of the sensors using enough number of sensors in most cases provides more than 90% initial coverage of the ROI [14]. Obtaining a high percentage of coverage during the initial deployment is unreasonable as this level of uniformity for the initial deployment is impossible for most of the practical applications. Therefore, we aim to address this issue.

3) A long process from chaos to order state: In DSSA, the partial force is the key concept. The partial force in a single-point-deployment is large at the few first steps of the algorithm, which causes chaos in the system. The chaos situation proceeds to a stable state as the effect of it decreases by lower partial force. In the DSSA, the order state happens in the last steps of the algorithm before the full coverage is achieved. Therefore, the DSSA needs a long time to reach a stable state.

B. Self-Organising Deployment Algorithm (SODA)

Self-Organising Deployment Algorithm (SODA) is proposed to overcome the limitations mentioned earlier in DSSA. All those limitations are originated from the DSSA uniformly treating of any local area regardless of the density of each neighbourhood. Whereas, the partial force should depend on the density of each neighbourhood to adjust the intensity of applied partial forces in different circumstances. The number of sensors and expected density are two parameters that should determine the intensity of the applied partial force. Adjusting the applied force during the first few movements of each sensor is especially significance to avoid moving nodes chaotically. The partial force is stateless (has no information about the previous and next layout). Therefore, the number of sensors in the neighbourhood and the expected density at the end of the algorithm are used as a guide to control partial force to behave as it is expected.

The detail of this algorithm is presented in Figure 2. In SODA, two steps are considered: initialisation step and force calculation step. The SODA initiates its process by initialising the P, C_R, S_R, D, and calculating $\mathcal{M}$, which is the expected density. After initialisation, SODA is executed at each node as long as the node is unstable. An unstable node is defined as a

 ISBN: 978-1-61208-676-7

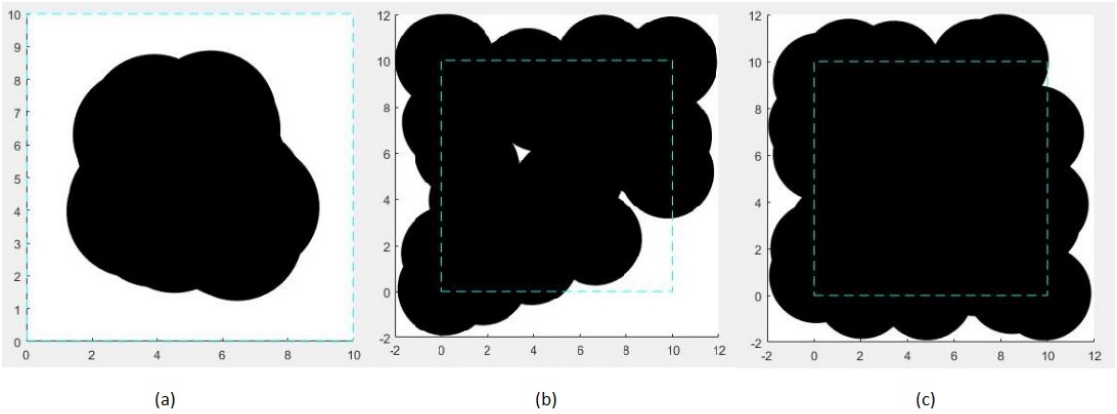

Figure 3. Covered area in a 10x10 region, 30 sensor nodes, $C_R = 4$ and $S_R = 2$: (a) Initial deployment. (b) Final coverage by DSSA. (c) Final coverage by SODA.

node in which has not been reached its oscillation or stability point.

The partial force at each node is calculated based on the local density at each neighbourhood. Equation 1 is used when the local density is lower than the expected density. The following equation is used otherwise:

$$Fnew_n(i,j) = \frac{D}{\mathcal{M} \times N}(C_R - |p_n^i - p_n^j|)\frac{p_n^j - p_n^i}{|p_n^j - p_n^i|} \quad (2)$$

where the applied force is reduced in the dense neighbourhood. In this equation, D, M, and N are local density, expected density and the number of neighbours, respectively. C_R stands for communication range, and p shows the positions of the sensors.

IV. Performance Evaluation

A. Simulation Specifications

The DSSA and SODA algorithms are simulated in a 10 x 10 region using Matlab. The C_R and S_R are assumed 4 and 2, respectively. The $threshold$ for oscillation and stability is considered to be 0.1522, the same as what is used in DSSA performance study [14]. The sensor nodes are considered to be randomly distributed around a point, which is chosen randomly in the ROI. The initial coverage for each scenario is different as the deployment point is chosen randomly. The final coverage by running DSSA in one scenario is presented in Figure 3(b), where this algorithm covers 97.2% of the area. These results are based on 30 sensor nodes, $C_R = 4$ and $S_R = 2$ in a 10 x 10 area. In this deployment, the whole coverage of the requested area is not achieved. The final coverage by SODA for the same initial deployment is presented in Figure 3(c). The initial coverage of 40.76% , presented in Figure 3(a), has resulted in a full coverage by SODA.

The final coverage and the mean travelled distance by every sensor are measured for the different number of deployed sensors in two areas: a 10 x 10 small area and another 20 x 20 area. To obtain reliable results, every experiment is repeatedly executed a number of times for every chosen deployment point, and the average results are taken. The value of $threshold$ for oscillation and stability in 20 x 20 area is considered to be half of the value of $threshold$ in a 10 x 10 area, which is 0.0761. This is because the $threshold$ should be smaller in larger areas since the same number of sensors needs more accurate movements in a larger area to be able to cover the area more efficiently.

B. Results

1) Area Coverage: The initial covered area and the final coverage of DSSA and SODA algorithms are shown in Figures 4 and 5, respectively. The initial coverage for both algorithms is the same because the initial deployment in both experiments is identical. As expected, the final coverage area provided by each algorithm is increased by increasing the network size (i.e., number of sensors). However, the results from the figures show an improvement in the covered area obtained by SODA compared to the DSSA's results in both scenarios. This difference decreases as the number of sensors increases. Therefore, the improvement in coverage descends as the number of sensors increases. It applies to both scenarios. In SODA a dense initial deployment can be locally recognised and very close sensors can be separated gradually regardless of the initial percentage of the coverage. Therefore, the coverage provided by SODA is 10% increased in sparse scenarios where the network is not crowded with sensors. The effectiveness of coverage in SODA in larger networks is more noticeable than in smaller networks. As can be seen from Figure 4, SODA continuously performs well and can achieve up to 20% higher coverage than that of DSSA in an area of 20 x 20.

2) Mean distance: Figures 6 and 7 show the mean distance travelled by every node in both DSSA and SODA. The total distance travelled by every node before reaching a stable state is not appropriate for comparison. Therefore, the mean distance is calculated to compare DSSA and SODA from this perspective. The mean distance is important in case of power usage, and movement mobility of every sensor, which at last causes network stability. Figures 6 and 7 show that the SODA has a smaller mean distance and in result uses less power generally in both areas. The correct movement of the sensors in SODA decreases the transition time from chaos to order state in comparison to DSSA algorithm. The applied partial force in SODA considers the local density and sensor numbers that

 ISBN: 978-1-61208-676-7

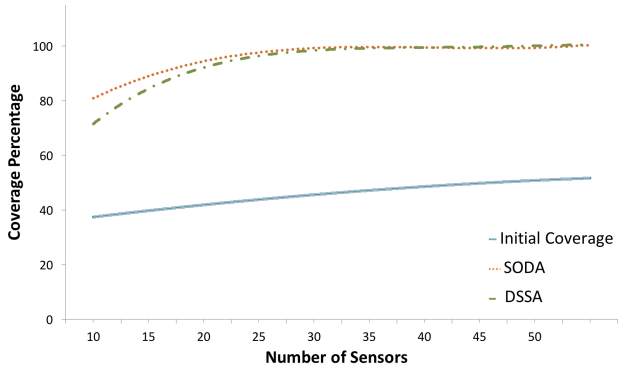

Figure 4. The area coverage in 10 x 10 region

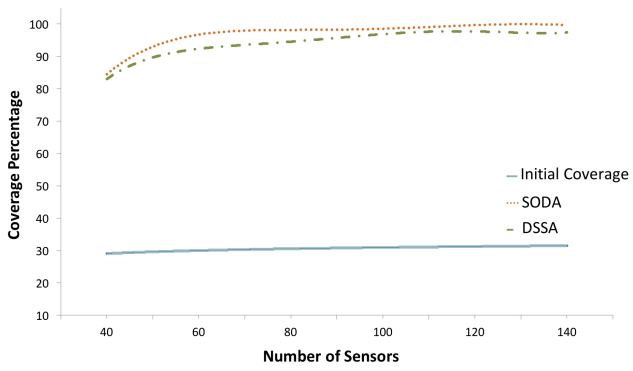

Figure 5. The area coverage in 20 x 20 region

causes more appropriate movements. The proper movement causes the sensors to reach their steady state sooner.

The behaviour in Figures 6 and 7 show that by increasing the number of sensors, the mean distance that every sensor travels rises and very slowly decreases after a while. In Figure 8, three different number of sensors are simulated in the SODA algorithm, and this image is captured after three runs. Based on Figures 6 and 7, the mean distance that every sensor travels increases as the number of sensors increases and then decreases after a while. The turning point in Figure 6, for instance, is 110 sensors. The result of the simulation in Figure 8 shows 6.52, 7.36, and 6.53 as mean distance of 150, 115, and 80 sensors respectively. This data confirms the result from Figures 6 and 7.

The theoretical reason behind this behaviour is the distribution of the sensors. In any size of the area, the mean distance of every sensor node increases as the number of sensor increases. The increase of sensor numbers leads to the higher partial force for every sensor node that causes more movements. However, this increment behaviour stops after a certain point, which is called the Optimal Number of Sensor (ONS). The ONS is where the full coverage of an area is achieved. Although, before ONS point the mean distance of sensors increases, the reduction in mean distance is seen after this point. This behaviour is also based on the density of sensors after this point. The number of neighbours for every sensor increases as the total number of sensors increases. The distribution of these neighbours is asymmetric before ONS point. It makes partial force to be large; however, when the number of sensors is greater than the ONS, the node arrangement in every neighbourhood tends to be more symmetrical and hence, their forces counterbalance every other. Consequently, less force implied shorter movement, which results in a lower mean distance.

V. Conclusion and Future Works

Coverage can be considered as the practical measurement of wireless sensor networks due to its direct impact on the network performance. Many factors, such as sensors technical specifications, network topology and most importantly, deployment algorithms influence the designated coverage. Maximising coverage using the resource constrained nodes is usually the goal of any deployment algorithm.

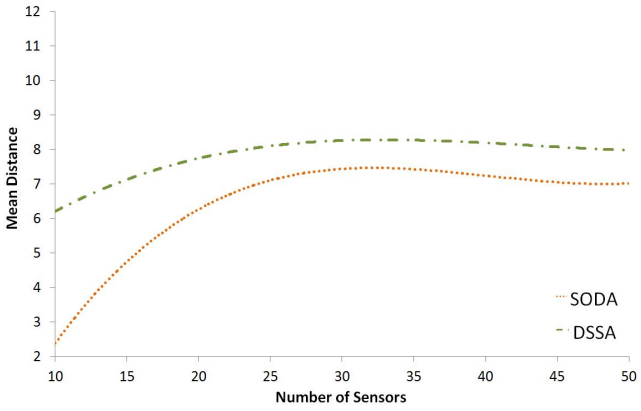

Figure 6. Mean distance in 10 x 10 region

In this paper, we have proposed SODA for mobile sensor networks. SODA is based on the DSSA that is reported in the literature, which is inspired by the equilibrium of molecules to provide the required coverage. However, the effectiveness of DSSA is highly dependent on the initial deployment of the nodes and also subject to providing a minimum initial coverage. These issues have been addressed in SODA. Furthermore, SODA's transition from chaos (i.e., initial deployment) to order (stabilised nodes) state is faster.

In our performance study, the performance of SODA has

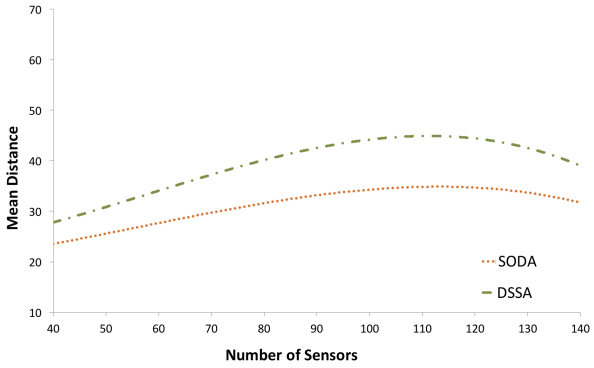

Figure 7. Mean distance in 20 x 20 region

 ISBN: 978-1-61208-676-7

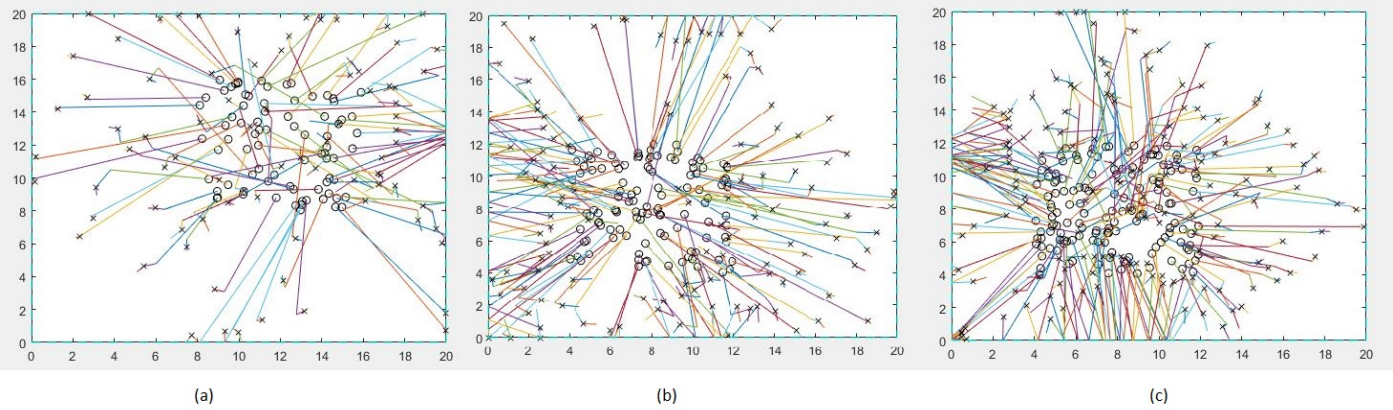

Figure 8. Sensors movements in 20x20 region: (a) Movements of 80 sensors. (b) Movements of 115 sensors. (c) Movements of 150 sensors.

been compared to DSSA, by simulating both algorithms in Matlab and measuring the final percentage of the coverage, and the mean distance travelled by every node. Simulation results confirm the advantages of SODA to achieve a more uniform distribution of nodes after applying the algorithm and hence a better coverage. The SODA solution has improved the final percentage of coverage by 10% and the mean distance has been reduced by 10% to even 60% in comparison to those of DSSA. The faster transition from chaos to order causes the faster symmetric form of each neighbourhood with less mean distance for every sensor and consequently resulting in lower power consumption.

In more realistic scenarios, WSNs can be used in a large area, where the ROI can be divided into multiple sub-regions for easy deployment. Finding an optimal solution to provide a trade-off between the number of sub-region and the designated coverage can be a direction for future work. Additionally, extending SODA to be able to utilise two-hops neighbouring information, or even more, when calculating the partial force at each neighbourhood may yield benefits beyond those of one-hop SODA. As another line, we are going to study this in the future.

REFERENCES

[1] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," Computer networks, vol. 52, no. 12, 2008, pp. 2292–2330.

[2] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: A survey," Computer networks, vol. 54, no. 15, 2010, pp. 2688–2710.

[3] G. Tuna, V. C. Gungor, and K. Gulez, "An autonomous wireless sensor network deployment system using mobile robots for human existence detection in case of disasters," Ad Hoc Networks, vol. 13, 2014, pp. 54–68.

[4] D. Tao and T.-Y. Wu, "A survey on barrier coverage problem in directional sensor networks," IEEE sensors journal, vol. 15, no. 2, 2015, pp. 876–885.

[5] T. J. Chowdhury, C. Elkin, V. Devabhaktuni, D. B. Rawat, and J. Oluoch, "Advances on localization techniques for wireless sensor networks: A survey," Computer Networks, vol. 110, 2016, pp. 284–305.

[6] A. Ghosh and S. K. Das, "Coverage and connectivity issues in wireless sensor networks: A survey," Pervasive and Mobile Computing, vol. 4, no. 3, 2008, pp. 303–334.

[7] A. Simonetto and G. Leus, "Distributed maximum likelihood sensor network localization." IEEE Trans. Signal Processing, vol. 62, no. 6, 2014, pp. 1424–1437.

[8] B. Horling, R. Vincent, R. Mailler, J. Shen, R. Becker, K. Rawlins, and V. Lesser, "Distributed sensor network for real time tracking," in Proceedings of the fifth international conference on Autonomous agents. ACM, 2001, pp. 417–424.

[9] A. Baranzadeh and V. Nazarzehi, "A decentralized formation building algorithm with obstacle avoidance for multi-robot systems," in Robotics and Biomimetics (ROBIO), 2015 IEEE International Conference on. IEEE, 2015, pp. 2513–2518.

[10] J. Liang and Q. Liang, "Design and analysis of distributed radar sensor networks," IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 11, 2011, pp. 1926–1933.

[11] V. Nazarzehi, A. V. Savkin, and A. Baranzadeh, "Distributed 3d dynamic search coverage for mobile wireless sensor networks," IEEE Communications Letters, vol. 19, no. 4, 2015, pp. 633–636.

[12] H. Mahboubi, K. Moezzi, A. G. Aghdam, K. Sayrafian-Pour, and V. Marbukh, "Distributed deployment algorithms for improved coverage in a network of wireless mobile sensors," IEEE Transactions on Industrial Informatics, vol. 10, no. 1, 2014, pp. 163–174.

[13] C.-F. Cheng, T.-Y. Wu, and H.-C. Liao, "A density-barrier construction algorithm with minimum total movement in mobile wsns," Computer Networks, vol. 62, 2014, pp. 208–220.

[14] N. Heo and P. K. Varshney, "Energy-efficient deployment of intelligent mobile sensor networks," IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 35, no. 1, 2005, pp. 78–92.

[15] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in Proceedings of the 7th symposium on Operating systems design and implementation. USENIX Association, 2006, pp. 381–396.

 ISBN: 978-1-61208-676-7