# Randomized Robot-assisted Relocation of Sensors for Coverage Repair in Wireless Sensor Networks

Greg Fletcher, Xu Li*, Amiya Nayak, and Ivan Stojmenovic
SITE, University of Ottawa, Canada
Email: gflet062@uottawa.ca, {xuli, anayak, ivan}@site.uottawa.ca

***Abstract*—In wireless sensor networks (WSN), stochastic node dropping and unpredictable node failure greatly impair coverage, creating sensing holes, while locally redundant sensors exist. If sensors are all equipped with locomotion, they will be able to relocate themselves to improve coverage. But this approach increases the complexity of hardware design for sensors as well as deployment budget. In this paper, we consider a small group of mobile robots to serve WSN. We propose an algorithm, named *Randomized Robot-assisted Relocation of Static Sensors* (R3S2), for coverage repair and a grid-based variant, called G-R3S2. By these algorithms, mobile robots move within the network to collect redundant sensors and deliver them to reported sensing hole positions. In R3S2, robots move completely at random and relocate encountered redundant sensors. In G-R3S2, the robots random movement is restricted on a virtual grid, and the robots continually move to the next least recently visited grid point so as to increase the chance of discovering redundant sensors and sensing holes. Through extensive simulation, we show their effectiveness and practicality and evaluate their performance. The simulation results indicate in particular that G-R3S2 outperforms R3S2 across all measured metrics.**

## I. INTRODUCTION

A wireless sensor network (WSN) consists of a massive number of micro-sized, cheap sensing devices powered by low-energy batteries and connected by wireless communication links. It is often randomly deployed (e.g. by aircraft) in a region of interest (ROI) for surveillance purpose. And it is expected to cover the ROI continuously and maximally throughout its lifetime so as to provide consistent, quality sensing service. For a WSN where there is no control on sensor placement, it is not possible to obtain and maintain a maximized hole-free coverage in the presence of stochastic node dropping and unpredictable node failure.

Recently, controlled mobility has been introduced for improving coverage in WSN. Each sensor is attached to certain mobile platform, e.g. robot or unmanned vehicle, and obtains the ability to geographically relocate. After their initial random placement, such mobile sensors self-deploy and achieve an improved coverage over the ROI [6]; as the network evolves, redundant sensors autonomously relocate to fill emerging sensing holes [8]. However, considering the large-scale nature and the low deployment budget of the network, equipping every sensor with locomotion is not cost-effective. Therefore, in this paper we envision using a small group of mobile robots (thus less hardware investment required) to serve a WSN for coverage improvement.

Only a few robot-assisted sensor deployment schemes [1], [3], [9] were proposed in the literature. Robots are assumed to have sensors in their 'hand' all the time. They move around in the ROI according to some policies and place sensors to construct a connected WSN with the desired coverage [1], [3]. When sensing holes occur (due to node failure), they move to fill the holes with spare sensors [9]. However, it may not be physically feasible that a few robots carry the entire set of network nodes. In some cases, the sensor network may have already been deployed, and robots are supposed to reuse the deployed sensors from an economic point of view. Thus these existing algorithms have very limited applicability.

We consider, for the first time to use bare robots to improve coverage of an existing random WSN. In this case, robots serve as network maintainer. They improve existing coverage by transferring redundant sensors to reported sensing hole positions. The delay from the moment when a sensing hole occurs to the moment when it is filled ought to be minimized. In our parallel work [2], this robot-assisted (also termed carrier-based) coverage repair problem is modeled as a new variant of the NP-complete Traveling Salesman Problem (TSP) or Vehicle Routing Problem (VRP), depending on the number of robots used. It is One-Commodity TSP with Selective Pickup and Delivery (1-TSP-SELPD) if there is only one robot, and One-Commodity VRP with Selective Pickup and Delivery (1-VRP-SELPD) otherwise. The uniqueness is that the demand of any delivery customer (i.e., sensing hole) can be met by a relatively large number of pickup customers (i.e., redundant sensors). We proposed the first centralized solution to 1-TSP-SELPD by ant colony optimization [2].

While seeking for centralized solutions to 1-VRP-SELPD, here we propose a localized algorithm, named *Randomized Robot-assisted Relocation of Static Sensors* (R3S2) and a grid-based variant (G-R3S2) to solve the above stated coverage repair problem. By this algorithm, robots move along a random trajectory; they collect encountered redundant sensors and carry them to fix encountered sensing holes. In G-R3S2, robots movement is restricted on a virtual grid and instructed by neighboring sensors. Each sensor records the number of times that every adjacent grid point is visited by robots; when a robot enters its communication range, the sensor recommends the grid point least often visited to the robot. Under sensors navigation, robots attempt to explore least visited areas in the network and therefore have an increased chance of discovering redundant sensors and sensing holes. Both R3S2 and G-

---

*Corresponding author

978-1-4244-3574-6/10/$25.00 ©2010 IEEE

R3S2 have very low message overhead because only local communication is involved. Through extensive simulation, we evaluate their performance on repair delay and message cost with varying number of sensors, holes and robots.

The rest of the paper is organized as follows. We first review related work in Sec. II. Then we define the network model in Sec. III and introduce a local sensing hole identification method in Sec. IV. After that, we present R3S2 and G-R3S2 in Sec. V and VI, respectively. Our simulation study is reported in Sec. VII, followed by the closing remarks in Sec. VIII.

## II. RELATED WORK

To the best of our knowledge, this paper is the first that investigates robot-assisted sensor relocation for coverage repair in a static WSN. Existing carrier-based sensor placement algorithms [1], [3], [9] require robots to always have spare sensors in their hand and construct/repair coverage only using those sensors. They are not sensor relocation algorithms. Existing sensor relocation protocols [8], [12] aim at a different problem, sensor self-relocation, and require all sensors to have locomotion (a strong requirement leading to high deployment cost). Due to space limitations, we will briefly review carrier-based sensor placement algorithms below, and refer readers to our recent survey article [7] for sensor relocation algorithms.

Batalin and Sukhatme [1] presented a single-robot-based sensor placement algorithm. The robot carries all the sensors and moves randomly step by step. It drops a sensor after each step if its current position is not occupied by any sensor. Deployed sensors monitor the movement of the robot and record the robot's coming and going directions. The robot consults with neighboring sensors for a direction least recently visited and moves accordingly. Fletcher et al. [3] presented a back-tracking based algorithm supporting multiple robots. Robots deploy sensors at vertices of a virtual grid constructed over a bounded ROI. They move along the grid to an open direction in a pre-defined order and, when obstructed, back track to the last-visited entrance to unexplored area.

Mei et al. [9] presented three robot coordination protocols for coverage repair. In a centralized protocol, a robot is appointed central manager and handles sensor failure reports. Each robot keeps updating the central manager with its latest position. Sensors report detected node failures to the central manager. The central manager dispatches robots to replace the reported failed sensors with spare sensors carried by the robot. In a distributed protocol, the sensory field is partitioned into sub-regions, and the centralized algorithm is run in each sub-region. In a dynamic protocol, a Voronoi diagram is dynamically constructed using robots based on hop count. Nodes report detected sensor failures to the robots of their home Voronoi cells. The robots then move to replace the failed sensor with spare sensors that they are carrying.

## III. MODEL AND DEFINITIONS

We consider a WSN randomly deployed in the ROI, whose boundaries are known a priori. A continuous uncovered area in the ROI is called *sensing hole* (hole for simplicity). Although the ROI may not be fully covered, some sensors could be redundant from a local perspective. To save energy, these sensors stay in "sleeping" mode, and are thus called *passive* sensors. Non-redundant sensors are essential for coverage. They remain alert and are thus referred to as *active* sensors.

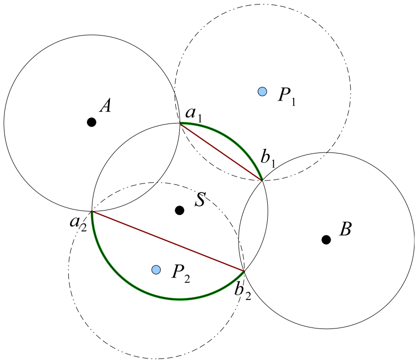


Fig. 1. Local sensing hole identification

Sensors have the same communication radius $r_c$ and the same sensing radius $r_s$. We assume $r_c \geq 2r_s$. There are some communicable mobile robots randomly scattered in the ROI. They have unique ID and the same communication radius $R_c$, which is not smaller than $r_c$. Direct communication can take place between any two nodes, whether sensors or robots, only when they are within each other's transmission range.

An activity scheduling algorithm [4] is adopted for determining passive sensors. By this algorithm, sensors determine their own status, passive or active, using one-hop neighborhood information only. Before falling "asleep", passive sensors choose a closest active neighbor as *proxy* by sending a delegation message to it. Nodal relative position can be used as a tie breaker for proxy selection.

The goal is to develop an effective and efficient sensor relocation algorithm, by which mobile robots find and collect passive sensors and deliver them to reported holes. Here, "efficient" means fast hole healing and minimal communication.

## IV. LOCAL SENSING HOLE IDENTIFICATION

An active sensor $S$ learns the positions of active neighbors by periodic "hello" messages. Through local computation it identifies the arcs on its sensing range perimeter that are not covered by any active neighbor. Any uncovered arc that is larger than, or equal to, degree $\pi/2$, will be divided evenly in half. Arc division is carried out, at most 3 times in the worst case that the initial arc has a degree of $2\pi$, until all uncovered arcs have a degree not larger than $\pi/2$.

Finally, each uncovered arc implies an adjacent sensing hole. $S$ will consider that the number $N_h(S)$ of adjacent holes is equal to the number $N_a(S)$ of uncovered arcs. In reality, $N_h(S)$ can be less than $N_a(S)$ (even without intentional arc division) as two local holes are possibly part of the same large hole. This inaccuracy is inevitable in a localized algorithm.

For a local hole defined by an uncovered arc, its position (i.e., center) is defined as the position symmetric to the location of $S$ by the chord of the arc. For example, in Fig. 1, $S$ has two uncovered arcs $a_1b_1$ and $a_2b_2$ (thickened) on its sensing range perimeter; it decides that there are two holes respectively at $P_1$ and $P_2$. The intuition is that, if an additional sensor $S'$ is placed there, the hole will be exactly filled with minimum sensing range overlapping between $S$ and $S'$.

## V. ALGORITHM R3S2

Robots travel within the ROI step by step autonomously and asynchronously at random; they do not communicate while moving. For every robot $R$, there is a *discovery phase* of predefined length between any two successive movement steps. In this phase, $R$ remains static, and periodically transmits a beacon message carrying its current location. On receiving the beacon message, nearby active sensors reply with the locations of their adjacent sensing holes as well as delegated passive sensors (if any exists).

Let $\mathcal{H}$ and $\mathcal{V}$ respectively be the hole set and the passive sensor set discovered by robot $R$ during a discovery phase. Denote by $\mathcal{S}$ the set of sensors currently held by $R$. We use $\Phi$ to imply the emptiness of a set. At the end of the discovery phase, $R$ makes a decision on its next movement, either continuing random movement (in this case the robot is said to be *free*) or moving to fulfill a task (in this case, the robot is said to be *busy*), i.e., picking up a passive sensor and/or fixing a hole, according to the following policies:

1) In the case of $\mathcal{H} = \Phi$ and $\mathcal{V} = \Phi$,
   there is no task for $R$, and $R$ decides to continue its random movement.
2) In the case of $\mathcal{H} \neq \Phi$ and $\mathcal{V} \neq \Phi$,
   $R$ chooses to repair a hole $H \in \mathcal{H}$ with a sensor $V \in \mathcal{V}$ so that $|RV| + |VH|$ is minimized.
3) In the case of $\mathcal{H} \neq \Phi$ and $\mathcal{V} = \Phi$,
   $R$ chooses to repair a closest hole $H \in \mathcal{H}$ if $\mathcal{S} \neq \Phi$, or continue its random movement otherwise.
4) In the case of $\mathcal{H} = \Phi$ and $\mathcal{V} \neq \Phi$,
   $R$ decides to continue random movement with probability $1 - p$ and to pick a closest sensor $V \in \mathcal{V}$ with probability $p$, where $p = 1$ if $\mathcal{S} = \Phi$, and $\frac{1}{|\mathcal{S}|}$ otherwise.

In Case 4 the purpose of probabilistic pickup is to increase the chance that every robot discovers a passive sensor and fixes its encountered holes. After a busy robot finishes its task, or when the task is canceled (see below), the robot starts a new discovery phase immediately at its current location.

If two robots are neighboring each other during their discovery phase, they will discover each other and compete for any common task. If the competition is for picking up a passive sensor (in Case 4), the robot carrying a smaller number of passive sensors wins; if it is for repairing hole (in Case 3), the robot with a larger number of passive sensors wins; if it is for a pair of hole and sensor (in Case 2), the robot with shorter travel distance wins. The winner proceeds as usual, while the loser removes relevant objects (sensors or holes) from consideration and re-makes its movement decision.

Every time when a robot picks up a passive sensor, it informs the corresponding proxy to remove the sensor from further consideration. If the target sensor does not exist anymore (e.g., due to the pickup by another robot), a robot simply cancels its task. Having arrived at the target hole, a robot tries to re-discover the hole by listening to hello messages of neighboring sensors. It drops a sensor if the hole still exists, and cancels the task otherwise. A relocated passive sensor becomes active and broadcasts a 'hello' message to declare its arrival. All active sensors receiving the Hello message reassess local holes. As soon as a robot accomplishes or cancels its task, the algorithms enters next iteration.

## VI. GRID-BASED R3S2

In R3S2, a robot attempts to traverse the ROI by random movement, and discover and handle all sensing holes and passive sensors encountered. The robot moves completely at random when it is free. Unlike R3S2, the grid-based version (G-R3S2) imposes some constraints on robot movement in order to lower randomness, shorten expected traversal time, and reduce coverage repair delay. Note that sensors are still randomly deployed as in R3S2.

G-R3S2 is the implementation (with improvement) of R3S2 on a virtual gird, where the movement of a free robot is restricted on the grid. Any grid can be used as long as by traversing it a robot is able to reach at least once the communication range of a sensor located in any point in the ROI. One example is square gird of edge length $\sqrt{2}r_c$; another example is equilateral triangular grid of edge length $\sqrt{3}r_c$. G-R3S2 translates the sensor and hole discovery problem in the continuous ROI into a traversal problem on the discrete grid.

At initiation, robots spontaneously align themselves with the grid by moving to a closest grid point. They then move asynchronously from grid point to grid point in steps. A busy robot will likely have to move off the grid to perform the given task. When the task is completed or canceled, the robot starts a discovery phase immediately if it is located at a grid point; otherwise, it has to first return to the previous grid point.

In addition to restricting the robots movement on a virtual grid, G-R3S2 further restricts robot movement by using a *Least Frequently Visited* (LFV) policy [1]. Simply speaking, a free robot is required to move to a least frequently visited adjacent grid point rather than to a randomly selected one. Random choice is made only in case of a tie or in the event that there are no active sensors within the robots' communication radius. The LFV-based grid point selection obviously helps remove randomness in network exploration and improves the algorithm performance further. We shall elaborate on it below.

Each robot locally maintains an increasing *sequence number* (SeqNo), which will be used together with its ID to define a distinct visit to a grid point. It increases this number whenever its residence grid point changes. An off-grid robot keeps its last visited grid point as its residence grid point so that its returning visit (due to task completion or cancellation) is not counted. During a discovery phase, a robot broadcasts a beacon message carrying its ID, SeqNo, and grid location, and the message is replied by active sensors as in R3S2.

For each of its neighboring grid points, an active sensor maintains a visit count and the last robot visit defined by the $(robot\_ID, SeqNo)$ pair. It increases the visit count when a distinct robot visit to the point is observed, i.e., when it receives from the point a beacon message containing a $(robot\_ID, SeqNo)$ pair different than the recorded last visit.

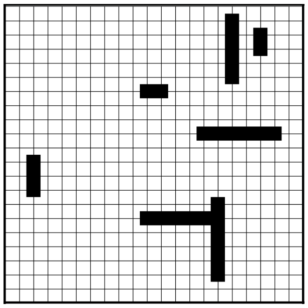


Fig. 2. Simulation environment. The gird is for robot movement in G-R3S2.

When replying to the beacon message of a robot, an active sensor includes the adjacent grid point with smallest visit count in the report message. As such, after a discovery phase, a robot is able to identify the least frequently visited neighboring grid point and then move to it if free.

## VII. PERFORMANCE EVALUATION

In this section we evaluate R3S2 and G-R3S2 through extensive simulation. We use the following performance metrics.

- Robot Moves (RM): The average number of movements made by each robot during the simulation.
- Total Moving Distance (MD): The average total distance traveled by each robot during the simulation.
- Bit Cost (BC): The average number of bits transmitted by each sensor replying to robot beacon messages.

Both RM and MD are indicators to per robot energy consumption and coverage repair latency. Recall that there is a discovery phase for each robot between two successive movement steps. BC reflects per sensor energy consumption for communication (due to the tested algorithms). While these metrics should be minimized, PR expects maximization.

### *A. Simulation Setup*

We implemented R3S2 and G-R3S2 with a custom network simulator. The sensors and robots are initially placed randomly throughout a rectangular ROI containing six obstacles of various shapes, sizes and configurations. We set $r_s = 12\sqrt{2}$, $r_c = 2r_s = 24\sqrt{2}$ and $R_c = r_c = 24\sqrt{2}$. In G-R3S2, we chose to use a virtual square grid of edge length equal to $r_c\sqrt{2} = 24$. Figure 2 shows the test ROI with the virtual grid drawn. The black blocks represent obstacles.

Passive sensors are identified by the activity scheduling algorithm [4]; sensing holes are determined locally, following the method outlined in Sec. III. Robots move asynchronously step by step. For each step, they move at a random speed between 0.1 and 1.1 and a distance of 24 (equal to the grid edge length). The length of the discovery phase between two movement steps is 100 simulated time units. The simulations terminate either once the robots have picked up all redundant sensors and none of the robots have any additional redundant sensors on hand or there are no more holes to fill.

To ease analysis, in simulation, we did not consider emerging holes caused by run-time sensor failure, although their treatment is automatically provided by the same algorithms. Thus the locations and number of the passive sensors and sensing holes was determined randomly based on the initial sensor layout. We first varied the number of robots, $m$, from 1 to 7, while fixing the number of sensors in the environment, $n$, to 400. We then varied $n$ from 300 to 500, in increments of 50 sensors, while fixing the $m$ to 3. We ran the simulation 20 times, with random sensor and robot distribution, for every scenario in order to obtain the average results.

### *B. Analysis*

The following is a detailed analysis of the results generated from the simulations of the proposed algorithms. As we will see G-R3S2 outperforms R3S2 across all measured metrics.

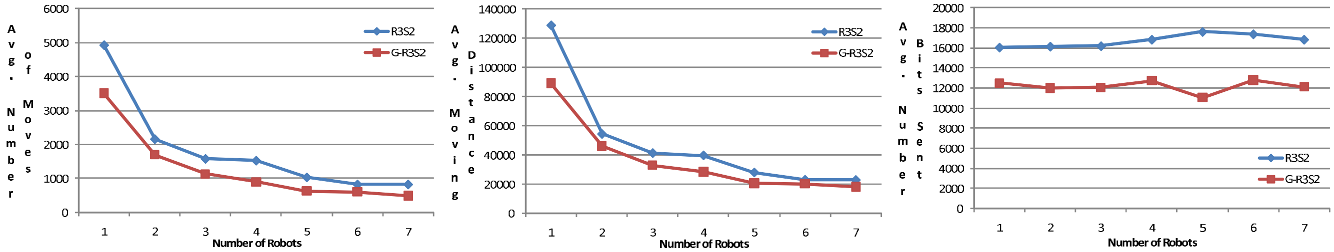
*1) Impact of number of robots:* We first study the impact of the number of robots, $m$, on the algorithm performance. Our simulation results for this purpose are shown in Fig. 3.

Figure 3(a) illustrates the relation of RM and $m$. We notice a decreasing trend in the curves for both R3S2 and G-R3S2. This confirms the intuition that the expected coverage repair delay decreases as there are more robots serving the network. We also notice that as the number of robots increases the difference in performance is minimized. The reason for the large difference in performance with fewer robots is that the grid-based algorithm is more intelligent, as the robots move to directions visited less frequently and thus areas more likely containing passive sensors and holes. But, as the number of robots increases, R3S2 increases its probability of finding passive sensors and holes despite the random movement strategy.

Figure 3(b) illustrates the relation between MD and $m$. As with RM, we notice a decreasing trend in both curves. Again we notice a significant difference in performance when there are 5 or fewer robots in the environment. As the number of robots increases, the difference in performance becomes less than the difference seen in Fig. 3(a). This is because G-R3S2 requires extra moving distance when the robots move from a grid point to either pick up or drop a passive sensor (at an off-grid location) and then again to return to its previous grid point. As a result of this extra distance traveled the difference between the performance of G-R3S2 and R3S2 with more than 5 robots is negligible in our simulation.

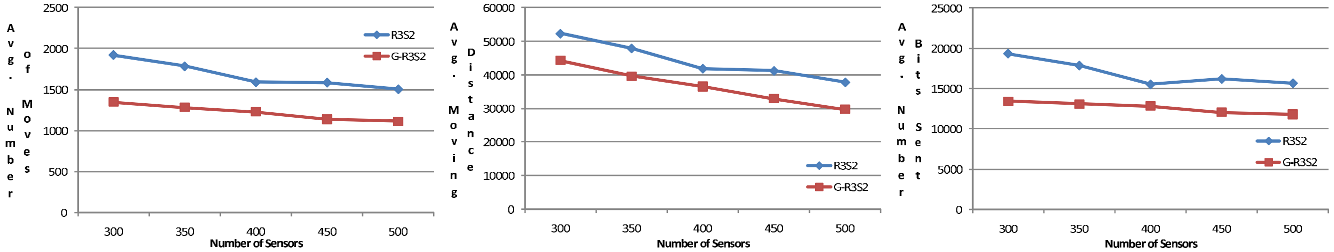
Figure 3(c) illustrates the relation between the BC and $m$. We notice a near flat trend for both R3S2 and G-R3S2. This flat trend is expected. BC depends on the number of sensing holes and passive sensors in the environment and the number of times they are suggested to robots (as reply to a discovery phase). When $n$ is fixed, the average number of holes and passive sensors is roughly fixed under random sensor distribution. The same sensing hole or passive sensor may be suggested more often when there are more robots. But the probability is very small or negligible as it is subject to the chance that multiple robots visit the same hole or sensor with random movement in a large area plus the change that the hole/sensor is not handled during previous robot visits. The difference in performance between the two algorithms is due to R3S2 requiring more movement increasing the number of discovery phases and thus the BC for the replies.

*2) Impact of number of sensors:* We now examine the impact of the number of sensors, $n$, on the algorithm per-



(a) Robot moves (RM) (b) Total moving distance (MD) (c) Bit cost (BC)

Fig. 3. Impact of $m$ on movement and bit cost



(a) Robot moves (RM) (b) Total moving distance (MD) (c) Bit cost (BC)

Fig. 4. Impact of $n$ on movement and bit cost

formance. Our simulation results are shown in Fig. 4.

In Fig. 4(a) and 4(b) we observe the relation between RM and MD with $n$, respectively. In both figures we notice a monotonically decreasing trend for both algorithms. Once again G-R3S2 is seen to outperform R3S2, with the difference in performance near constant regardless of the number of sensors deployed. The reason for the decreasing trend is that – despite the stochastic nature of the sensor deployment – as the number of sensors in an environment increase, it is likely that the number of coverage holes will decrease and the number of passive sensors will increase. Thus with fewer holes to heal and more passive sensors to relocate the robots' movements and distance traveled decrease.

Figure 4(c) illustrates the relation between BC and $n$. For G-R3S2, we again notice a monotonically decreasing trend. For R3S2 we notice an overall decreasing trend with a larger than expected drop in bit cost for the $400$ sensor scenario. As previously discussed the number of bits transmitted will decrease with the number of robot movements.

## VIII. CONCLUSIONS

WSN is an active field full of challenging research issues [5], [10], [11]. In this paper we addressed novel carrier-based coverage repair and presented a localized solution, R3S2 and its grid-based variant G-R3S2. R3S2 repairs sensing holes using redundant sensors already deployed in the environment with the assistance of mobile robots. G-R3S2 is an implementation of R3S2 on a virtual grid with *Least Frequently Visited* policy for motion planning. It limits the amount of randomness in the robots' movements and is thus more efficient than R3S2.

As previously mentioned, to the best of our knowledge R3S2 and G-R3S2 are the first algorithms for relocation of static sensors for coverage repair, as existing coverage repair algorithms either require robots to initially carry sufficient spare sensors (no sensor relocation) or require each sensor to have locomotion (mobile sensor). We demonstrated R3S2 (and G-R3S2) using a single robot case, as well as multiple robot cases. Through simulation we showed that both R3S2 and G-R3S2 provide reasonable results across all measured metrics and that G-R3S2 performed better in general.

## ACKNOWLEDGMENT

This work was partially supported by NSERC Strategic Grant STPSC356913-2007B.

## REFERENCES

[1] M.A. Batalin and G.S. Sukhatme. "Coverage, Exploration and Deployment by a Mobile Robot and Communication Network". Telecommunication Systems, 26:181–196, 2004.

[2] R. Falcon, X. Li, A. Nayak, and I. Stojmenovic. "The One-Commodity Traveling Salesman Problem with Selective Pickup and Delivery: an Ant Colony Approach". In Proc. of IEEE CEC. 2010. To appear.

[3] G. Fletcher, X. Li, A. Nayak, and I. Stojmenovic. "Back-Tracking Based sensor Deployment by a Robot Team". In Proc. of IEEE SECON, 2010.

[4] A. Gallais, J. Carle, D. Simplot-Ryl, and I. Stojmenovic. "Localized Sensor Area Coverage with Low Communication Overhead". IEEE Tran. on Mobile Computing, 7(5): 661–672, 2008.

[5] F. Ingelrest, D. Simplot-Ryl, and I. Stojmenovic. "Optimal transmission radius for energy efficient broadcasting protocols in ad hoc and sensor networks". IEEE TPDS, 17(6):536-547, 2006.

[6] X. Li, H. Frey, N. Santoro, and I. Stojmenovic. "Focused Coverage by Mobile Sensor Networks". In Proc. of IEEE MASS, pp. 466-475, 2009.

[7] X. Li, A. Nayak, D. Simplot-Ryl, and I. Stojmenovic. "Sensor Placement in Sensor and Actuator Networks". Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication, Wiley, 2010.

[8] X. Li, N. Santoro, and I. Stojmenovic. "Mesh-based Sensor Relocation for Coverage Maintenance in Mobile Sensor Networks". In Proc. of UIC, LNCS 4611, pp. 696-708, 2007.

[9] Y. Mei, C. Xian, S. Das, Y.C. Hu, and Y.-H. Lu."Sensor replacement using mobile robots". Computer Comm., 30(13):2615-2626, 2007.

[10] I. Stojmenovic. "Localized network layer protocols in sensor networks based on optimizing cost over progress ratio". IEEE Network, 20(1):21-27, 2006.

[11] I. Stojmenovic, D. Liu, and X. Jia. "A scalable quorum based location service in ad hoc and sensor networks". Int'l Journal of Communication Networks and Distributed Systems, 1(1):71-94, 2008.

[12] G. Wang, G. Cao, T. La Porta, and W. Zhang."Sensor Relocation in Mobile Sensor Networks". Proc. IEEE INFOCOM, pp.2302-2312, 2005.