EURASIP Journal on
Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

MSNS: mobile sensor network simulator for area coverage and obstacle avoidance based on GML

Young-Sik Jeong[1], Youn-Hee Han[2], James J Park[3*] and SooYoung Lee[4]

Abstract

A mobile sensor network is a distributed collection of sensors, each of which has sensing, computation, communication, and locomotion capabilities. In particular, locomotion facilitates the ability to self-deployment. In such a network of self-deployable mobile sensors, it is difficult to evaluate the effectiveness of mobile sensor network deployment in a given target area because we cannot predict the coverage rate for the target area. The coverage rate will be changed due to the number of sensor required in the target area, connectivity degree to be maintained and unknown obstacles. In this article, we develop mobile sensor network simulator (MSNS) in order to visualize (1) coverage secured by mobile sensors and (2) avoidance of obstacle objects (building, road and wall, and so on) on the real map drawn by GML (Geography Markup Language). From a user, MSNS receives the number of mobile sensor nodes, connectivity degree, sensor node's sensing range, communication range, and supersonic wave range. And then it visualizes the location information of sensor nodes, connectivity degree, and sensing coverage, all of which change with simulation time. Thereby we can estimate how many nodes are required in a given target area, and also calculate coverage rate of the target area in advance to the real deployment of mobile sensors.

Keywords: mobile sensor network, visual coverage, connectivity, potential field

1. Introduction

Mobile sensor network is made up of group/groups of small low-power sensor nodes that can sense specific situations or collect information, and then transmit that information to sink nodes using wireless ad hoc communication. In general, mobile sensor network, which is very useful for the target fields to be difficult to access, should be constructed by using mobile sensor nodes with sensing, computation, communicating, and locomotion capabilities. In particular, locomotion facilitates the ability to self-deployment. Several nodes with various kinds of sensors for sound, heat, magnetic field, and infrared ray are randomly scattered in a target area. These sensors move, voluntarily avoiding obstacles and other nodes, establish sensing coverage and configure their communication network [1]. And after sensing the information, the sensor transmits such information as sensing information to the sink node through routing path. The sink node sends the sensing information to middleware or server before processing it for application. This technology is used in various fields such as medical care, transportation, military, environment, and disaster prevention.

Coverage and connectivity are ones of critical factors to establishing mobile sensor network [2,3]. The coverage means the area in which sensing by sensor nodes is possible. The connectivity means how many sensors are connected to cover the entire area for sensing or detecting, and deliver any sensing information to the sink node. The mobile sensor network, established in a given target area where terrain status is unknown, is required to maximize sensing coverage with mobile sensors and maintain the connectivity as much as a network administrator requires.

When self-deployable mobile sensors are deployed in a given target area to be required for monitoring, sensing, and detecting; however, it is difficult to predict how many sensors are needed in the target area and how much connectivity the sensor network have, which prevents guaranteeing the effectiveness of network deployment.

* Correspondence: jhpark1@snut.ac.kr
[3]Department of Computer Science and Engineering, Seoul National University of Science and Technology, Seoul, South Korea
Full list of author information is available at the end of the article

In this article, a new simulator, mobile sensor network simulator (MSNS), is designed and implemented. From a user, MSNS receives the information on the number of mobile sensor nodes, connectivity degree, sensor node's sensing range, communication range, and supersonic wave range. In the simulator, the target area is set when a user designates a random area where obstacles are set by using GML (Geography Markup Language). A number of sensors input by the user are randomly deployed in the target area, and they move to avoid obstacles while maximizing the coverage in the target area and maintaining the given connectivity. MSNS visualizes the location information of sensor nodes, connectivity degree, and sensing coverage, all of which change with simulation-time. It can be used to find out the coverage rate of the target area secured by the given mobile sensors. Thereby we can estimate how many nodes are required in a given target area, and also calculate coverage rate of the target area in advance to the real deployment of mobile sensors.

This article is organized as follows. Section 2 presents the existing simulators related to this research, moving method of sensor in the mobile sensor network, and the results of studies on coverage and connectivity of sensors. Section 3 explains the GML-based obstacle setting technique, which is the one of keys to MSNS, and the MSNS coverage algorithm to be used for self-deployment of mobile sensors. Section 4 shows the design and implementation of MSNS based on the technique suggested in Section 3. Section 5 presents the evaluation of MSNS's functions and the results of coverage algorithm's performance. Finally, Section 6 suggests the conclusions and discussion on the future researches.

2. Related studies

Related studies are explained in the two perspectives: development of ubiquitous sensor network (USN) simulator and coverage and connectivity of mobile sensor network. First, existing USN simulators focus on the verification of packets, protocol, and the network. Through such a method, a simulation can be run on the network lifetime on some simulators. TOSSIM, an open source TinyOS-based simulator from UC Berkeley, can simulate Mica2 series simulation from CrossBow. Main features are packet loss calculation and CRC sensing. However, it can only work with Mica2 series. GloMoSim, a PARSEC (C-based parallel simulation language)-based discrete event simulator, is a simulation environment for wireless mobile network. Like OSI 7-hierarchical model, GloMoSim is composed of number of layers. It monitors packet transmission status, and verifies network model or transmission scenario; however, it cannot work as sensor network. GloMoSim's next version, QualNet is a massive wireless network simulator. It uses IEEE 802.11 MAC and Physical Layer standard, and like GloMoSim it has several layers. When modules for layer are developed by different designers, the scenarios and models are being tested. Packet flow statistics can be checked through automatically collected data from each layer. Features for sensor network are designed as well; nevertheless, visualization of sensed objects. NS2 is most widely used network simulator, and many wired and wireless network simulators have been developed based on this system. It is a discrete event simulator, it can simulate various network protocols; however, it has too many nodes and is difficult to adapt to complex massive system. It also has too much unnecessary interdependency. J-Sim is a JAVA-based open source WSN simulator. Each component uses autonomous component architecture, and imitates software with IC chips. It is designed in loosely coupled structure so that is can support plug & play. It can calculate memory usage, number of events, and running time according to size of given network. It also simulates transmission status of transmitted event from target node being transmitted to sink nodes in packet form. Nevertheless J-Sim is difficult to visualize target node sensor. SWANS is an expansion of Jist, a PARSEC-based scattered event simulator. It is an open source simulator, and compared to NS2 or GloMoSim, it can carry out massive network simulation; nevertheless, like other simulators it can only carry out protocol verification [4,5].

Second, suggestion was made on the algorithm that enabled maximization of the area that could be covered in the mobile sensor network where potential field was applied so that initial sensors moved voluntarily [6]. On the assumption that there was a repulsive force between sensors or sensor and obstacle, such force was used to have sensors dispersed evenly on the network and to ensure that friction force, opposite to the repulsive force, was applied so that sensors reached the static equilibrium without any movement. The algorithm suggested in this article basically utilizes what is suggested in [6]. However, the difference is that sensors are induced in the way that local coverage is maximized rather than sensors simply being spread.

The previous studies [7] suggested self-deployment algorithm where Voronoi diagram was used. First of all, they raised the question if sensors were enabled to observe detection area at the maximum while minimizing move time, moving distance of sensor and complexity of message for a random detection area. In order to solve such problem, the studies suggested that it was necessary to find out coverage hole that was not observed by using sensor and to properly move sensor to enable observing the coverage hole. To this effect, three methods were suggested including VECtor-based algorithm (VEC), VORoni-based algorithm (VOR), and

Minimax method. In each method, information on location of neighboring sensors is acquired for each step, Voronoi polygon is drawn, and then, sensors move in the way to minimize the area where coverage is not secured in such polygon. Among the three methods, the VOR shows that sensors move toward the most distant vertex of the Voronoi polygon while the Minimax shows that sensors move toward the circumcenter of the Voronoi polygon.

As the case with [7], the previous study [8] also solved the problem on self-deployment of sensors based on the Voronoi diagram. The study [8] suggested the method that sensors utilized information on location of neighboring sensors for each step to configure the Voronoi polygon before moving toward the centroid of the polygon. The centroid of the Voronoi polygon is the mean position of all points inside the Voronoi polygon. In other words, the centroid is the point that a random sensor has the smallest value in the sum of variance of distance up to each vertex of the Voronoi polygon. If sensor moves to this point, the sensor is placed in the best position to cover the Voronoi polygon [9].

Lastly, the previous study [10] suggested that sensors should be moved in consideration of not only minimization of moving distance, but also remaining energy because self-deployment of sensors in itself consumes a great deal of energy. The study presented three algorithms that ensured the balanced deployment of the entire network by changing the degree of movement in consideration of local density of each sensor (number of neighboring sensors) and remaining energy. Table 1 shows comparison of the characteristics of the existing methods and the MSNS.

3. Mechanism of field and mobile sensor moving

3.1. Field establishment

Establishment of the target area is one of the important issues to execute MSNS. When mobile sensor network is established for various applications, the number of terrains is as many as the number of applications. Therefore, MSNS uses the method of establishing field where mobile sensor network is to be formed and based on such terrain, selecting target area that requires observation.

MSNS uses the GML [11-14] to establish field that includes obstacles. Since the GML is the standard for geospatial data, it has the high compatibility and is convenient for configuring field. And as the GML contains the coordinate information, it is possible to utilize the information to calculate the actual coordinate information of mobile sensors that estimate the locations of each other based on the relative coordinates. In the GML, factors that can be obstacles such as building are written mostly with polygon. Therefore, MSNS sets the polygon of the GML as an obstacle and processes it.

3.2. Coverage of MSNS

Another important issue to implement the MSNS is the moving technique for mobile sensors. The mobile sensors are required to maintain a given connectivity, avoid obstacles, and maximize coverage in the target area. Therefore, this article suggests the coverage method that adds the obstacle avoidance method to the constrained coverage method that maximizes coverage while maintaining the given connectivity [8,15].

The method has preconditions as follows. First, the method is based on the binary model that mobile sensors sense the target within the sensing range at the rate of 100% but cannot sense the target out of the sensing range. Second, all of the mobile sensors have the equal sensing distance (R_s) and the equal communication distance (R_c). Third, mobile sensors have the method to determine their location in order to calculate virtual force. Lastly, the method does not take into consideration distortion of sensing range and communication range of mobile sensors due to waves reflected by obstacles.

Table 1 Moving method of mobile sensors

Characteristics	Potential [7]	Constrained coverage [8]	VEC [10]	VOR [10]	Minimax [10]	Centroid [4]	Floor [2]	MSNS
Basic Strategy	**Repulsive power between sensors**	**Repulsive power between sensors and gravitation for keeping *k*-connect**	**Repulsive power between sensors**	**The first moves to Voronoi vertex**	**Move to circumcircle of Voronoi vertex**	**Move to center of Voronoi vertex**	**Move to floor line with at least overlap of sensing area**	**Repulsive power between sensors, gravitation for keeping *k*-connect repulsive between obstacle and sensor**
Coverage Type	**sensor coverage**	**sensor coverage**	**sensor coverage**	**sensor coverage**	**sensor coverage**	**sensor coverage**	**sensor coverage**	**sensor coverage**
Connectivity	x	o	x	x	x	x	o	o
Asynchronous	o	o	x	x	x	x	x	o
Avoid obstacle	o	x	Δ	x	x	x	o	o
Control message	x	x	x	x	x	x	O	x

O, support; Δ, partial support; x, not support.

Based on the potential filed method [6] frequently used for movement of robot in mobile robotics, the three virtual forces such as F_{cover}, F_{degree}, and F_{obstacle} are used for movement of mobile sensors. In order to maximize coverage, mobile sensors are basically required to have a certain distance from one another to ensure that their sensing range is not overlapped with others. F_{cover} is the force with which mobile sensors push against one another to maximize the sensing range in the target area. $F_{\text{cover}}(i, j)$ means the force that the sensor S_i takes from the neighboring sensor S_j during the unit time, which is expressed in Equation (1).

$$F_{cover}(i,j) = \frac{-C_{cover}}{\Delta_{ij}^2} \cdot \frac{x_i - x_j}{\Delta_{ij}} \tag{1}$$

In Equation (1), x_i and x_j represent the locations of sensors s_i and s_j while Δ_{ij} means the Euclidian distance of sensors s_i and s_j. And C_{cover} is the constant that means force of field.

In mobile sensor network, the mobile sensor that senses the information that requires observation in the target area uses the connection between mobile sensors in order to send the collected information to sink node. In this case, if a parent sensor on the path that is used to send information to one sink node loses connection for reasons such as failure or malfunction, a child sensor uses an alternative sensor that exists within its communication distance to form a new path. This local connectivity influences the entire connectivity [3]. In addition, sensors are required to maintain communication with a certain number or more of their neighboring sensors in order to deploy numerous mobile sensors in the target area with some sensors in the active state and others in the sleep state, which aims at increasing lifetime of the network [16].

F_{degree} is the force that is exerted by mobile sensors to keep the number of given neighboring sensors at the degree K. Figure 1 shows deployment of sensor nodes in case of $K = 3$. If the number of neighboring sensors is larger than K that should be kept, F_{degree} does not take place, and sensors become distant from each other due to F_{cover}. The sensors become more distant gradually to maximize coverage, and if the number of neighboring sensors is equal to a given degree K, F_{degree} takes place. As a result, a sensor draws its neighboring sensors to keep the number of neighboring sensors at the given degree K. $F_{\text{degree}}(i, j)$ means the force that the sensor S_i takes from its neighboring sensor S_j during the unit time, which can be expressed in Equation (2).

$$F_{degree}(i,j) = \begin{cases} \frac{-C_{degree}}{(\Delta_{ij} - R_c)^2} \cdot \frac{x_i - x_j}{\Delta_{ij}} & \textit{if count of neighbor sensor} = k \\ 0 & \textit{otherwise} \end{cases} \tag{2}$$

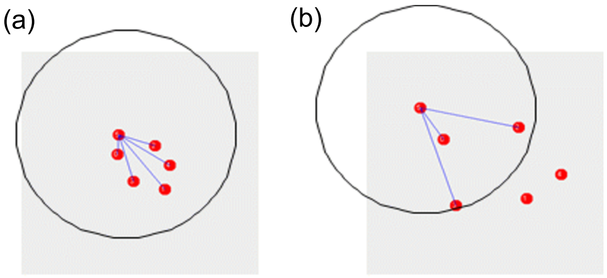


Figure 1 Deployment of sensors with K = 3. (a) Number of neighbor sensors > K; **(b)** number of neighbor sensors = K.

where R_c means communication distance while C_{degree} is the constant that means force of field. Mobile sensors in MSNS are required to maximize coverage, maintain the given connectivity and avoid obstacles. To this effect, this article defines F_{obstacle}.

It is assumed that mobile sensors are equipped with 16 supersonic wave sensors (sender = 8, receiver = 8) in order to obtain F_{obstacle}. Figure 2 shows that supersonic wave sensors locate obstacles. It is assumed that if supersonic wave distance (R_w) is determined and a sensor detects obstacle within the distance of R_w, the sensor with sensing range of R_w is located in the point that is two times of the distance between the obstacle and the sensor. F_{obstacle} is calculated in the same way as F_{cover} in order to maximize the range of R_w. $F_{\text{obstacle}}(i, k)$ takes place between the sensor node S_i and the obstacle o_k during the unit time, which can be expressed in Equation (3).

$$F_{obstacle}(i,j) = \frac{-C_{obstacle}}{\Delta_{ik}^2} \cdot \frac{x_i - o_k}{\Delta_{ik}} \tag{3}$$

where o_k is the location of obstacle while Δ_{ik} is the Euclidian distance between mobile sensor and obstacle, and C_{obstacle} is the constant that is caused by obstacle.

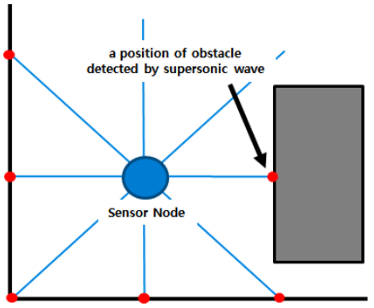


Figure 2 Location of obstacle with supersonic waves.

F_{cover}, F_{degree}, and F_{obstacle} are used to calculate the total virtual force F that mobile sensor S_i takes during the unit time, which is expressed in Equation (4).

$$\mathrm{F} = \sum\nolimits_{neighbors\ i}\left(F_{cover}(i, j) + F_{degree}(i, k)\right) + \sum\nolimits_{founded\ obstacles\ k} F_{obstacle}(i, k) \tag{4}$$

Mobile sensors continue to move at a constant speed if only F_{cover}, F_{degree}, and F_{obstacle} are considered. The virtual force F_{damper}, which needs to stop sensors from continuous movement, is defined as in Equation (5).

$$F_{damper} = \sigma \cdot v_{current} \tag{5}$$

where σ is damper constant while v_{current} is moving speed of the current sensor. F_{damper} is calculated based on the damper constant and the moving speed of the current node as shown above. For application of F_{cover}, F_{degree}, and F_{obstacle}, it is required to calculate C_{cover}, C_{degree}, and C_{obstacle} that are used for calculating each force. Mobile sensors push against each other to maximize coverage. And when the distance between them is $2R_{\text{s}}$, the coverage becomes the highest. In other words, when the distance between the two sensors is $2R_{\text{s}}$, F_{cover} does not take place between them, which is expressed in Equation (6). And C_{cover} can be calculated.

$$\left\| F_{cover} - F_{damper} \right\| = 0, \quad where\ \Delta_{ij} = 2R_s \tag{6}$$

In a similar way, when the distance between obstacle and sensor is R_{s}, F_{obstacle} does not take place to sensor, which is expressed in Equation (7). And C_{obstacle} can be calculated.

$$\left\| F_{obstacle} - F_{damper} \right\| = 0, \quad where\ \Delta_{ij} = R_s \tag{7}$$

Sensors move due to F_{cover}, F_{degree}, and F_{obstacle}. And there exists a moment when such forces reach the static equilibrium and the forces become zero (Equation 8).

$$\left\| F_{cover} + F_{degree} + F_{obstacle} - F_{damper} \right\| = 0, \quad where,\ \Delta_{ij} = \mu \cdot R_c \tag{8}$$

$$\alpha_{next} = \frac{\left(F - F_{damper}\right)}{m} \tag{9}$$

$$v_{next} = v_{current} + \alpha_{next} \cdot t \tag{10}$$

$$x_{next} = x + v_{next} \cdot t + \frac{1}{2} \cdot \alpha_{next} \cdot t^2 \tag{11}$$

In this case, μ is a safety factor, and the range of value is $0 < \mu < 1$. If the value becomes close to 1, sensors are readily disconnected. This equation is used to calculate C_{degree}.

The calculated force F that one node takes per unit time can be used to calculate acceleration of mobile sensor and to calculate speed by using the acceleration value. And lastly, it is possible to calculate the next location that sensor moves to (Equations 9-11). The coverage technique algorithm in MSNS is shown as follows.

Algorithm 1. MSNS coverage algorithm

```
While (MNS is playing)
  get neighbor nodes of current node;
  for each neighbor node do
    get $F_{cover}$ & $F_{obstacle}$;
  end for
  if # of neighbor node < = K then
    for each neighbor node do
      get $F_{degree}$;
    end for
  end if
  sum virtual force F;
  calculate acceleration;
  calculate velocity;
  calculate next location;
  if next location is contained in obstacle or outside of target area then
    next location sets current location;
  end if
end while
```

4. Design of MSNS

4.1. MSNS architecture

MSNS consists of user interface, GML analyzer, map layer manager, map controller, node manager, target area manager, and viewer. The overall structure of the MSNS is shown in Figure 3.

The user interface provides interface where users can enter set values necessary to start the MSNS. After the GML analyzer makes analysis of GML document, it creates map objects before sending them to the map layer manager. The map layer manager plays a role in managing map objects provided by the GML analyzer. And it also has control function related to map invoked by user interface. The map controller has the function of magnification, reduction, enlargement of area, and movement of the map information delivered to the map layer manager. The node manager applies sensor setting information entered in user interface to sensors, and creates and operates the obstacles defined in the map layer manager, the target area defined in the target area manager, and interactive sensors. The target area manager sets and manages the target area that requires detection and sensing in the field set by the GML document. The viewer visualizes map objects of the map layer manager and mobile sensors of the node manager.

4.2. Function of MSNS components

4.2.1. User interface

The user interface can be divided into map and node interfaces. The detailed structure of the user interface is shown in Figure 4.

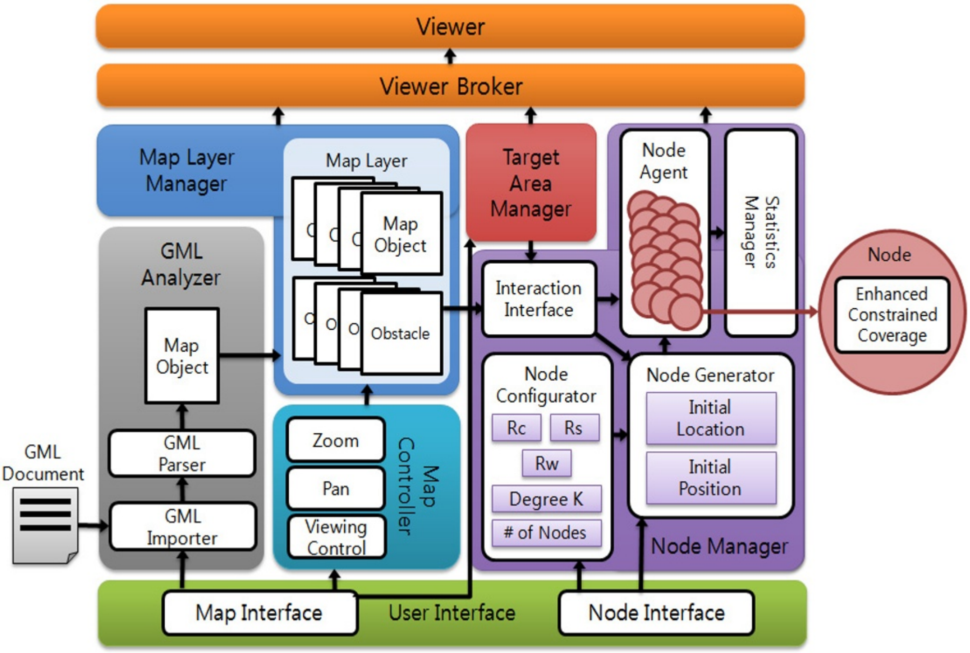


Figure 3 Overall architecture of MSNS.

The map interface consists of seven modules as follows: map chooser, which provides function of importing GML document to set the field where mobile sensor network is established; map lister, which visualizes the map, which is set up in MSNS, by object such as road or building; map extender, which magnifies the map; map reducer, which reduces the map; map mover, which moves the map to the desired place; extend area selector, which selects a specific area in the map and magnifies it; and target area selector, which selects the target area that requires detection and monitoring.

All of the functions set up by the map interface are delivered to the GML analyzer, the map controller, and the target area manager. Some of them can be used even after MSNS started deployment of mobile sensors. The node interface is composed of 16 modules. They include sensor adder, which adds mobile sensor to obtain desired coverage while MSNS is in operation; sensing range taker, which receives input of sensing range of sensor; communication range taker, which receives input of communication range of sensor; supersonic wave range taker, which gets input of supersonic

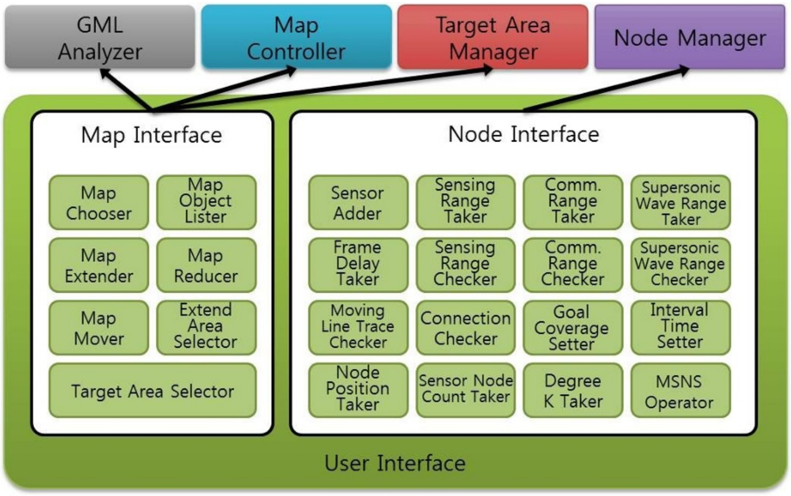


Figure 4 User interface architecture.

wave range of sensor to determine the location and distance of obstacle; frame delay taker, which gets input of the value for MSNS to adjust moving speed of sensor; sensing range checker, which provides the visualization information on sensing range of sensor; communication range checker, which provides communication range of sensor; supersonic wave range checker, which delivers supersonic wave range of sensor; moving line trace checker, which traces the distance in which mobile sensors moved; connection checker, which checks connectivity between mobile sensors; goal coverage setter, which stops operation of the MSNS when mobile sensors reach the desired coverage; interval time setter, which sets the time to pause operation of MSNS at a certain time interval; node position taker, which sets the method of deployment of mobile sensors in the beginning; sensor node count taker, which receives input of the number of sensors that will be spread in the target area of MSNS; degree *K* taker, which determines connectivity degree that sensors are required to maintain; and MSNS operator, which controls operation of MSNS. All of functions of node interface also can be changed while MSNS is in operation.

4.2.2. GML analyzer

Obstacle avoidance by using the GML in MSNS has the important meaning in the three perspectives. First of all, it is possible to simplify and visualize complex objects in the real world by using GML document. And it is easy to set obstacles because objects that can be obstacles in the GML such as building are configured in polygon. Lastly, the GML has the coordinate in the real world so that it is possible to take advantage of information on location of mobile sensor. The diagram of GML analyzer module is shown in Figure 5.

If the function to add a map through user interface is invoked, the GML document importer fetches GML document. The GML parser performs parsing of the GML document invoked by the GML document importer to extract map objects. The extracted map objects contain coordinate information, features, shape information, and information to determine whether or not such objects are obstacles. The map objects created by the map GML parser are delivered to the map layer manager.

4.2.3. Map layer manager

The overall diagram shows the relation between the map layer manager and other major modules of the MSNS in Figure 6. The map objects that come from the GML analyzer are transmitted to the map layer. The map layer distinguishes map objects from obstacles after examining whether or not the map objects are configured in polygon. Afterward, the obstacles are sent to the node manager. When the map objects and the obstacles are set up, the map layer manager calculates the boundary box of the map and sets the zoon ratio at 1.0. And it sets the map center point at the dead center point of the map. If the setting is completed this way, the map center point, the zoom ratio and the map boundary box are changed by the map controller, and the map is displayed on the MSNS through viewer broker.

4.2.4. Map controller

The map controller receives order of controlling the GML map and sets value to ensure that the map layer manager displays the map in the MSNS, following the request by user. The map center point setter sets value of the map's center in the map layer manager when the map is magnified/reduced and when the area of the map in interest is shifted. The zoom ratio configurator

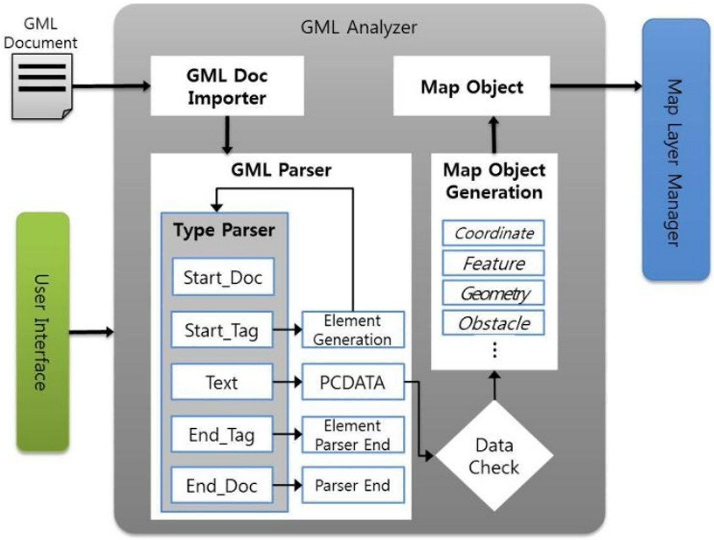


Figure 5 GML analyzer modules.

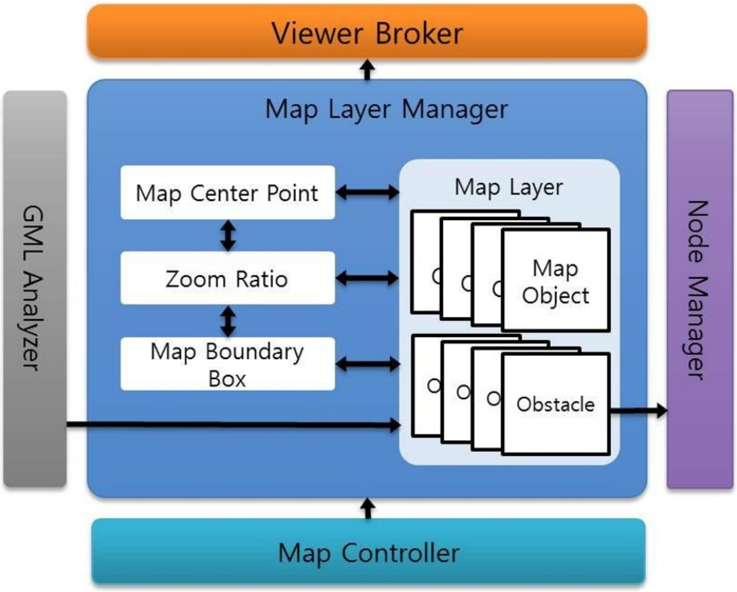


Figure 6 Relationship map layer manager with other modules.

sets the zoom ratio in the map layer manager when the map is magnified/reduced. The full extender is used when the magnified/reduced/shifted map is drawn again to the entire area. The overall diagram of map controller is shown in Figure 7.

4.2.5. Node manager

In Figure 8, we show the detailed structure of node manager. Node configurator saves the information on sensing range, communication range, supersonic wave range, degree of connectivity, and the number of sensors to be deployed. The saved values are applied in batch when sensors are created by node generator. The supersonic wave range is used to determine the location of obstacle in the target field where the sensor status is unknown. The node generator creates and manages the given number of sensors.

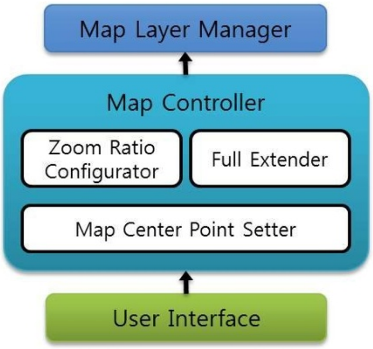


Figure 7 Map controller architecture.

Initial location of the node generator determines if sensors start moving from the corner of the given target area, from the center, or from a random position. In addition, based on the initial position, mobile sensors determine the location for them to be deployed within the initial location of the target area. Node agent manages mobile sensors and ensures that sensors move, avoiding obstacles in the target area based on the coverage algorithm suggested in Section 3, maintaining the degree K and maximizing the coverage. If force, which was generated by sensor to find the results that the sensors in MSNS moved to cover a proper target area, is equal to or less than a critical value, the node agent stops MSNS from operating. Interaction interface refers to the target area in the target area manager and refers to obstacles in the map layer manager. The initial location of sensors in the node generator is set outside of obstacles and outside of polygon. When sensors in the node agent move, they avoid obstacles in the target area.

Finally, statistics manager calculates information on mobile sensors observed by mobile agent. Elapsed time means unit time during which sensors move. Moving distance means the total distance that sensors move from their initially deployed position to the current position. Coverage rate provides information in percentage on the degree that mobile sensors cover the target area. Average neighbor means the average number of neighboring sensors out of the entire sensors.

4.2.6. Target area manager and viewer

The internal module diagram of target area manager and viewer are shown in Figure 9. The target area

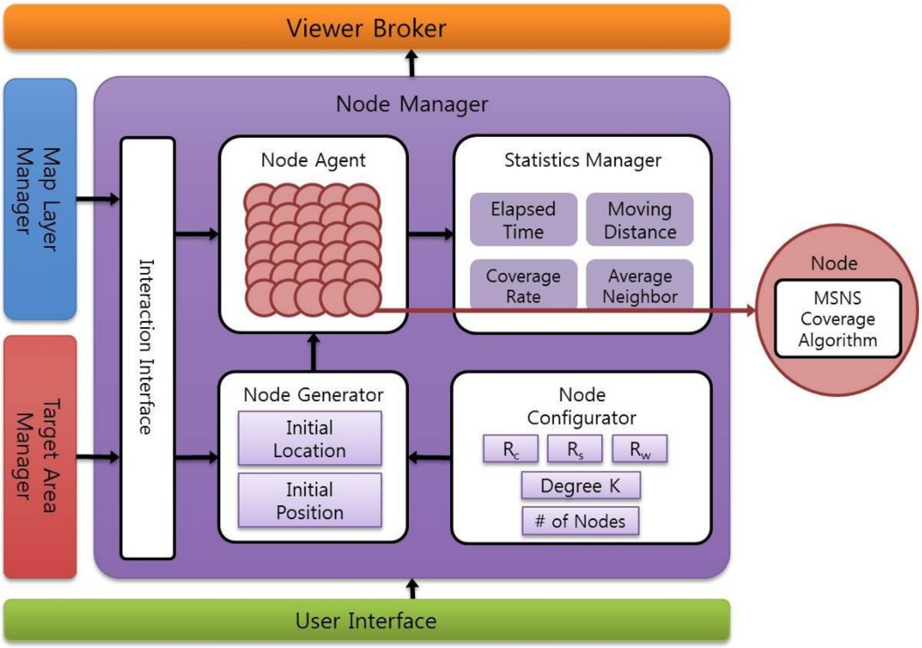


Figure 8 Node manager modules.

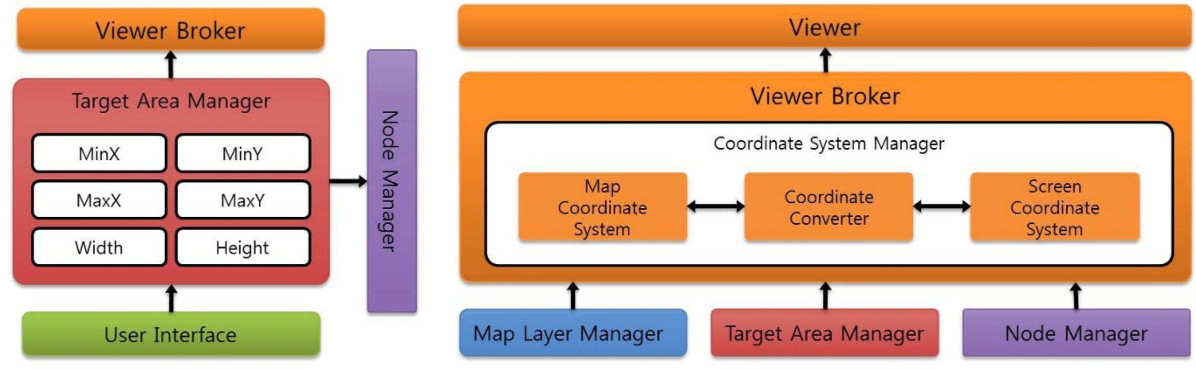
manager sets the target area in the mobile sensor network field. When a user drags the mouse to set the target area that requires observation, the manager saves and manages the coordinate. And it sends the information on the target area to the node manager to ensure that sensors are created and operated within the target area. Then, the information is sent to viewer broker in order to visualize the target area in MSNS.

Coordinate system manager of the viewer replaces target area of the target area manager and sensor of the node manager with MSNS screen coordinate based on the map coordinate that is set in the map layer manager. When map object is set by the map layer manager, map coordinate system is set by the boundary box. Screen coordinate system is set according to the size of view panel. If the screen coordinate system and the map coordinate system are set, the number of coordinate converters is calculated for conversion of the two systems, which enables a free conversion of MSNS screen coordinate and the GML map coordinate. The coordinate converter changes the screen coordinate to the map coordinate when a user sets the target area in the mobile sensor network field. And it changes the map coordinate to the screen coordinate when the map, the target area and the sensors are visualized in MSNS.

5. Implementation of MSNS

Figure 10 shows the initial screen and each control function of MSNS. Components of MSNS are as follows: toolbar; configuration panel, which has internal properties of sensor and provides information such as sensing coverage; status panel, which shows various information when MSNS is in operation; and viewer, which provides the GML coordinate information and status information of sensors. The toolbar consists of the add button to import GML document in order to configure the field of mobile sensor network, the map object lister button to classify the map provided by the GML document according to object, the zoom in button to magnify the map, the zoom out button to reduce the map, the full extension button to provide the map controlled by magnification/reduction/movement, the select zoom area button to select and expand a specific area on the map, the select target area button to set the target area on the map, the add sensor node button to add mobile sensor to ensure that the target area reaches the static equilibrium in the desired coverage when MSNS is in operation, and the position of sensor node button to show the information on location of the current mobile sensors.

Components of configuration panel are as follows: sensing range slide bar, which sets sensing range of sensor; communication range slide bar, which sets communication range of sensor; supersonic wave range slide bar, which sets supersonic wave range of sensor; frame delay slide bar, which adjusts moving speed of sensor; sensing range checker, which checks if the sensing range of mobile sensor is visualized; communication range checker, which checks if the communication range of sensor is visualized; supersonic wave range checker,

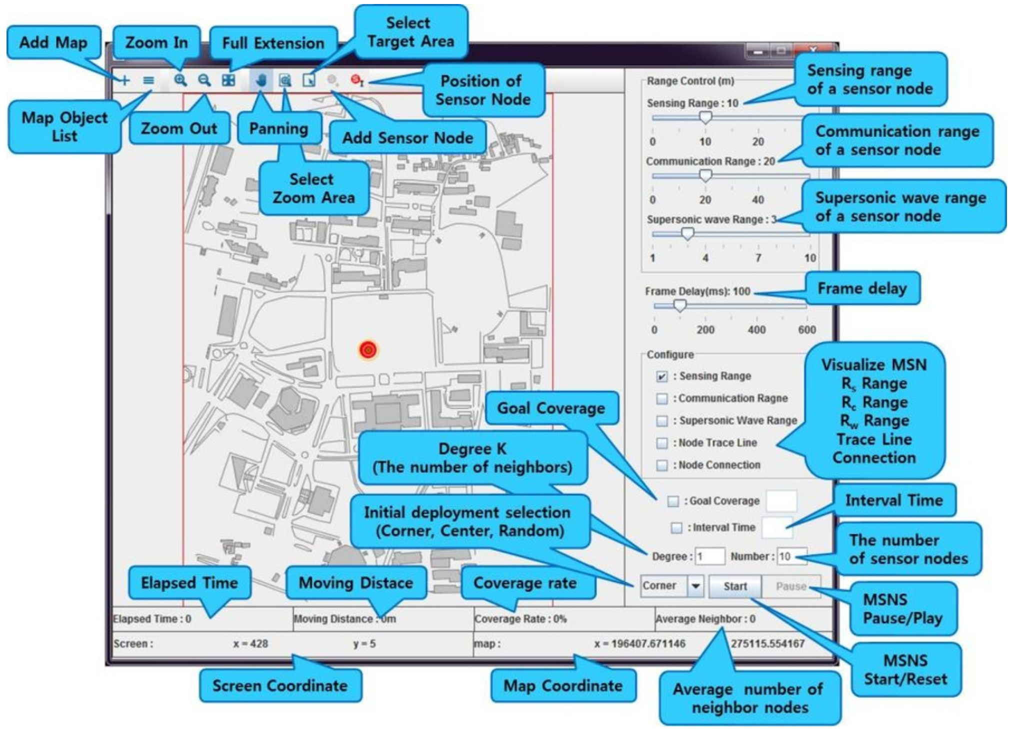


(a) Target Area Manager (b) Viewer

Figure 9 Module architecture of target area manager and viewer.

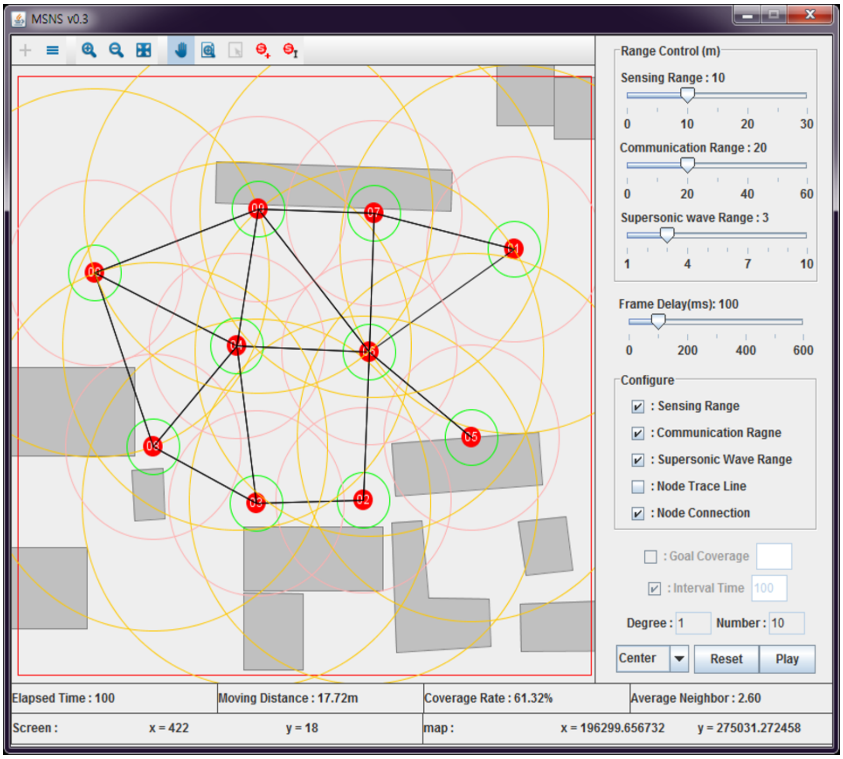
which checks if the supersonic wave range of sensor is visualized; node trace line checker, which checks if the distance in which sensor moved until the current time after being deployed in the beginning is visualized; goal coverage checker, which stops MSNS from operating when a user reaches the desired coverage; interval time checker, which pauses MSNS in operation during a certain time interval; degree *K* taker, which receives input of the number of neighboring sensors that sensors should maintain; number of sensor nodes taker, which receives input of the number of sensors to be deployed in the target area; initial deployment selector, which selects the initial location in which sensor are deployed when the MSNS starts; button to start and reset MSNS; and button to pause MSNS in operation and restart it.

Status panel is composed of elapsed time that represents simulation time of the current MSNS, moving distance that represents the total distance in which sensors move from the initial deployment location to the current location, following MSNS coverage algorithm, coverage rate that represents the degree that the target area is covered by the current sensors, and average neighbor



Range Control (m)
Sensing Range : 10
Communication Range : 20
Supersonic wave Range : 3
Frame Delay(ms): 100
Configure
☑ : Sensing Range
☐ : Communication Ragne
☐ : Supersonic Wave Range
☐ : Node Trace Line
☐ : Node Connection
☐ : Goal Coverage
☐ : Interval Time
Degree : 1 Number : 10
Corner
Elapsed Time : 0 Moving Distance : 0m Coverage Rate : 0% Average Neighbor : 0
Screen : x = 428 y = 5 map : x = 196407.671146 275115.554167

Figure 10 Implementation and panel functions of MSNS.



MSNS v0.3

Range Control (m)

Sensing Range : 10

Communication Range : 20

Supersonic wave Range : 3

Frame Delay(ms): 100

Configure

☑ : Sensing Range
☑ : Communication Ragne
☑ : Supersonic Wave Range
☐ : Node Trace Line
☑ : Node Connection

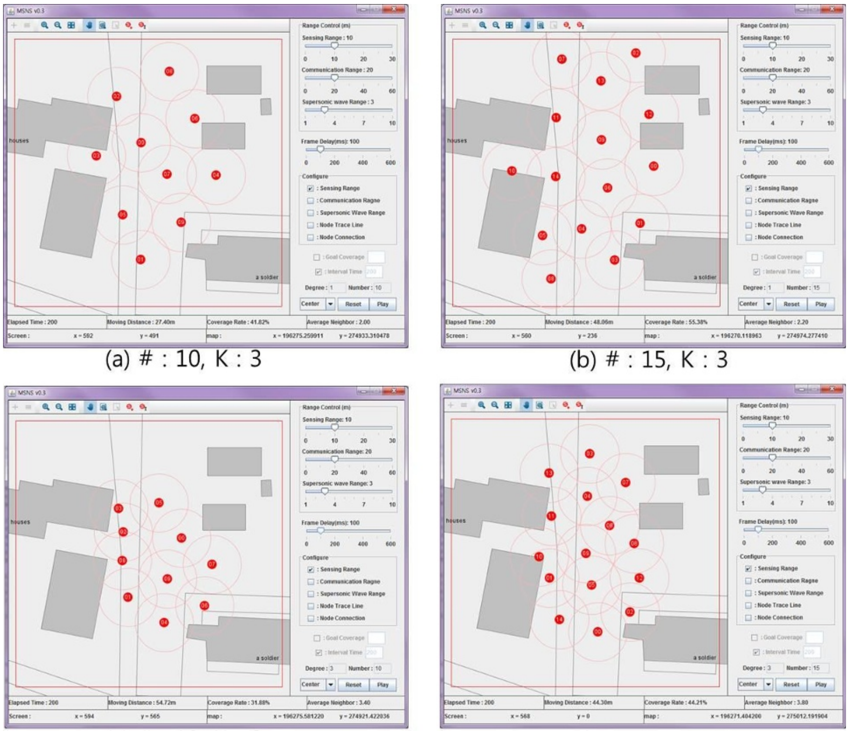
☐ : Goal Coverage
☑ : Interval Time 100

Degree : 1 Number : 10

Elapsed Time : 100 | Moving Distance : 17.72m | Coverage Rate : 61.32% | Average Neighbor : 2.60

Screen : x = 422 y = 18 map : x = 196299.656732 y = 275031.272458

Figure 11 Sensing, communication and supersonic wave range of MSNS.



(a) # : 10, K : 3

Degree : 1 Number : 10

Elapsed Time : 200 | Moving Distance : 27.40m | Coverage Rate : 41.82% | Average Neighbor : 2.00

Screen : x = 592 y = 491 map : x = 196275.259911 y = 274933.310478

(b) # : 15, K : 3

Degree : 1 Number : 15

Elapsed Time : 200 | Moving Distance : 48.06m | Coverage Rate : 55.38% | Average Neighbor : 2.20

Screen : x = 560 y = 236 map : x = 196270.118963 y = 274974.277410

(c) # : 10, K : 6

Degree : 3 Number : 10

Elapsed Time : 200 | Moving Distance : 54.72m | Coverage Rate : 31.88% | Average Neighbor : 3.40

Screen : x = 594 y = 565 map : x = 196275.581220 y = 274921.422036

(d) # : 15, K : 6

Degree : 3 Number : 15

Elapsed Time : 200 | Moving Distance : 44.30m | Coverage Rate : 44.21% | Average Neighbor : 3.80

Screen : x = 568 y = 0 map : x = 196271.404200 y = 275012.191904

Figure 12 Sensor deployment of MSNS with connectivity degree.

Table 2 Simulation of sensor deployment with connectivity degree

	Figure 12a	Figure 12b	Figure 12c	Figure 12d
Number of sensors	10	15	10	15
K	3	3	6	6
Coverage (%)	41.82	55.38	31.88	44.21
Average *K*	2.00	2.20	3.40	3.80

that represents the average number of neighboring sensors out of the total sensors.

Viewer provides the information, which is set up in the map toolbar, side configuration panel and base configuration panel, and the information on sensors that move to monitor the area depending on visualization.

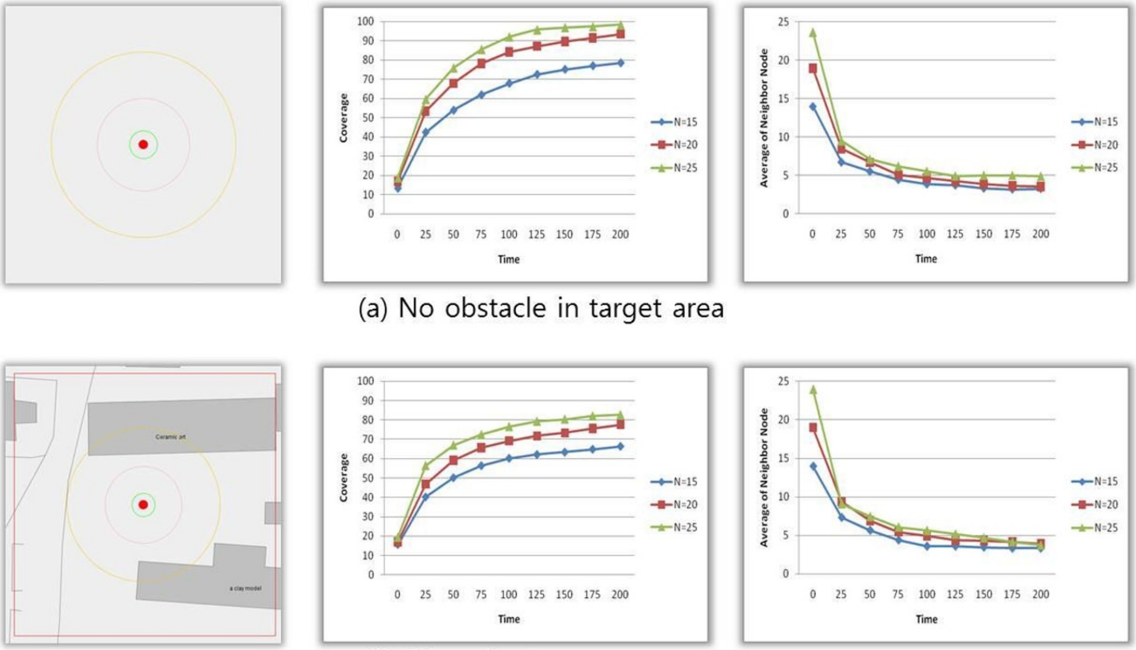
Figure 11 shows that map objects are expressed by using the GML and processed basically as obstacles. Since GML document keeps the information on large area, it is necessary to select the target area to deploy sensors. When the degree *K* of sensor is 1 and the number of sensors is 10, the figure shows deployment of sensors, sensing range and number of neighboring sensors that are maintained on average (pink circle: sensing range, orange circle: communication range, green circle: supersonic wave range). And the figure shows the sensing range of the target area in percentage (61.32%) along with the average (2.60) for each neighboring sensor out of all sensors.

6. Performance evaluation

For performance evaluation of the MSNS, it was assumed that sensors moved moderately if the evaluation was not based on time and if the average value of virtual force of mobile sensors deployed properly in the target area was smaller than the critical value. Then, the average value was used.

Figure 12 shows the connectivity degree of sensors in the same target area as the one where obstacles are set by the GML and the results of implementing MSNS after changing the number of sensors. Figure 12c, d shows the result that the sensing coverage of the target area diminished when simulation was conducted as the connectivity degree was increased while the number of sensors was kept constant in the same environment as the one in Figure 12a,b. In the different perspective, Figure 12b,d shows the result that the sensing coverage of the target area increased when simulation was conducted as the number of sensors was increased while the connectivity degree was kept constant in the same environment as the one in Figure 12a,c. The results of such simulation are shown in Table 2.

Figure 13a,b is the graphs that represent coverage, depending on number of sensors and elapsed unit time, and change in average number of neighboring sensors in the cases where obstacles did not exist and did exist. When the number of sensors was 15, 20, and 25, the



(a) No obstacle in target area

(b) Obstacle in target area

Figure 13 Number of sensors, coverage with time and average number of neighbor sensor (R_s = 10, R_c = 20, R_w = 3, corner, degree *K* = 1, Time = 0.1 s).

results of simulation were measured. As the time of simulation passed, the coverage increased gradually while the average number of neighboring sensors decreased.

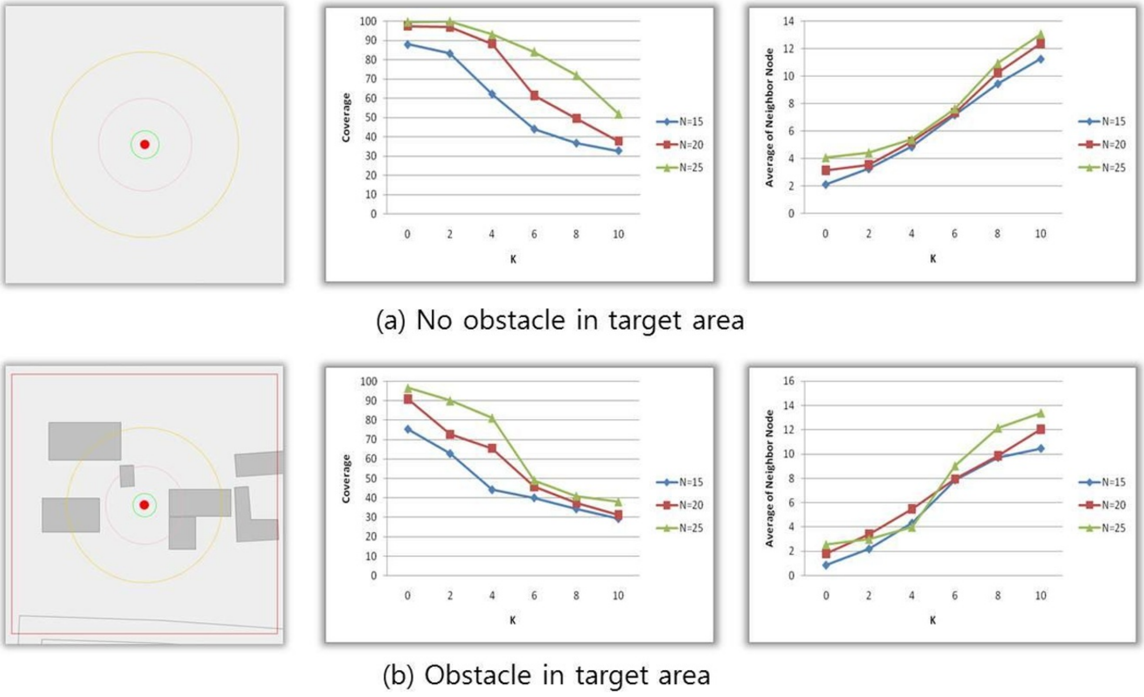
Figure 13b shows that the MSNS coverage algorithm operates in the similar way to that of the existing constrained coverage, avoiding obstacles.

Figure 14a,b is the graphs that represent coverage, according to degree K that should be maintained, and change in average number of neighboring sensors in the cases where obstacles did not exist and did exist. As the degree K increases, the coverage ratio of the target area decreases while the average number of neighboring sensors increases.

Figure 15a,b is the graphs that represent coverage, according to R_c and degree K, and change in number of neighboring sensors in the cases where obstacles did not exist and did exist. In case of $R_c > 2R_s$, the coverage ratio was higher while the average number of neighboring sensors due to the given degree K increased similarly. This result demonstrated that the size of R_c did not have the significant influence on the increase in average number of neighboring sensors.

7. Conclusion and future research

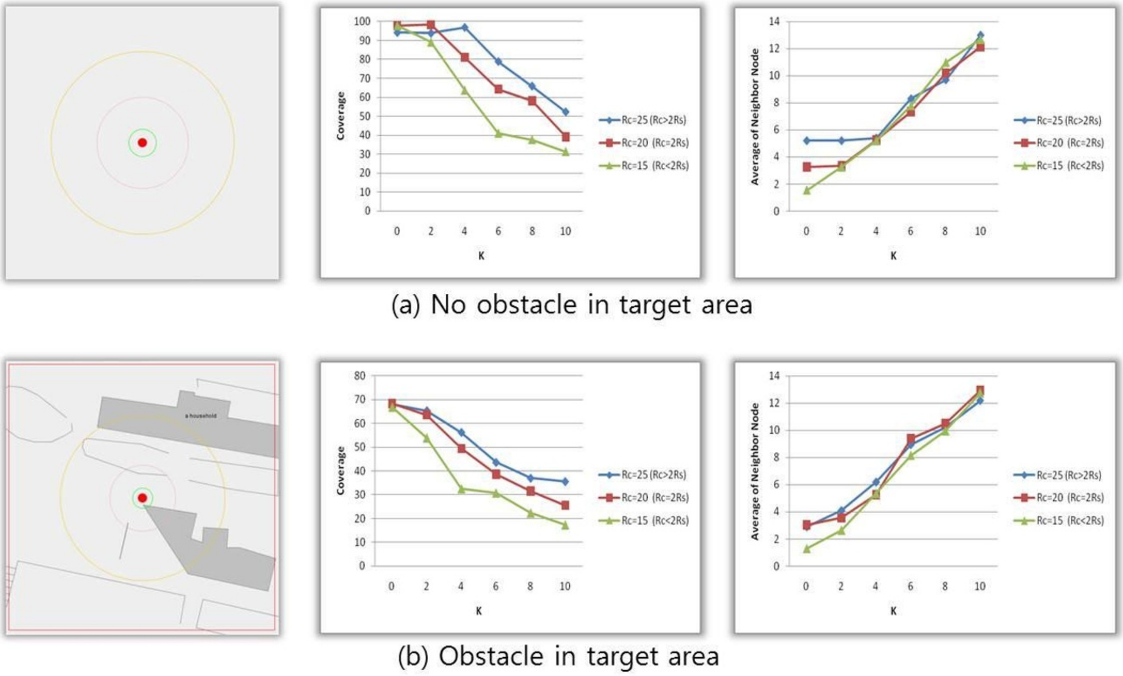
The MSNS developed in this article is the simulator that provides the information on sensing coverage of the target area where a number of mobile sensors are randomly deployed. Prior to the MSNS execution, the different connectivity degree is provided for different applications in consideration of the GML-based obstacles, which helps infer the number of sensors that are required in the target area and determine coverage ratio of the target area. In most of the previous studies, the connectivity is fixed or limited, or sensing range or communication range of sensor is determined in advance. In addition, other problems are that the information on location of sensors is displayed by using the screen coordinate, or shape of obstacles is determined in advance. MSNS uses the GML-based GPS coordinate information to utilize terrain coordinate. And based on such information, obstacles are set up so that it is possible to consider obstacles in various shapes. Furthermore, since the actual map coordinate is used, it is possible to estimate the actual coordinate of a sensor that moves on the actual map coordinate. It is also possible to conduct simulation of various environments since user interface is used to provide connectivity degree of sensors, sensing range, supersonic wave range, and number of sensors. The major contribution of this article is the proposal of a new virtual force to guide mobile sensors onto a more optimal path in terms of coverage expansion with respect to GPS and obstacle in theory aspect. This is achieved by incorporating the attractive force generated from the centroid of a sensor's local Voronoi



(a) No obstacle in target area

(b) Obstacle in target area

Figure 14 Number of sensors, coverage with degree *K* and variation of neighbor sensor (R_s = 10, R_c = 20, R_w = 3, corner, elapsed time = 200).



(a) No obstacle in target area

(b) Obstacle in target area

Figure 15 Number of sensors, coverage with degree *K* and variation of neighbor sensor (R_s = 10, R_c = 20, R_w = 3, corner, sensors = 20).

polygon with the repulsive forces generated by obstacles and neighboring nodes.

The future researches include applying probability model, instead of binary model, to sensing range and utilizing MSNS coverage algorithm for self-deployable mobile sensors with such different sensing range model. Another is to consider a more realistic possibility in distortion of sensing/communication range due to obstacles.

Acknowledgements
This research was supported by the IT R&D Program of MKE/KEIT [10035708, "The Development of CPS (Cyber-Physical Systems) Core Technologies for High Confidential Autonomic Control Software"] and also supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2011-0025975).

Author details
[1]Department of Computer Engineering, Wonkwang University, 344-2 Shinyong-Dong, Iksan 570-749, South Korea [2]Advanced Technology Research Center, Korea University of Technology and Education, CheonAn, South Korea [3]Department of Computer Science and Engineering, Seoul National University of Science and Technology, Seoul, South Korea [4]Embedded Software Research Department, Electronics and Telecommunications Research Institute, Daejeon, South Korea

Competing interests
The authors declare that they have no competing interests.

Received: 11 June 2011 Accepted: 8 March 2012
Published: 8 March 2012

References

1. R Arkin, K Ali, Integration of reactive and telerobotic control in multi-agent robotic systems, in *Third International Conference on Simulation of Adaptive Behavior, (SAB94) [From Animals to Animates]*, Brighton, England, pp. 473–478 (August 1994)
2. G Tan, SA Jarvis, A-M Kermarrec, Connectivity-guaranteed and obstacle-adaptive deployment schemes for mobile sensor networks. in *Proc of IEEE ICDCS 2008* 429–437 (June 2008)
3. F Xue, PR Kumar, The number of neighbors needed for connectivity of wireless networks. Wirel Netw. **10**(2), 169–181 (2004)
4. H-J Lee, Y-H Kim, Y-H Han, CY Park, Centroid-based movement assisted sensor deployment schemes in wireless sensor networks, in *2009 IEEE 70th Vehicular Technology Conference* (September 2009)
5. C-W Lee, S-W Kim, H-J Lee, Y-H Han, D-S Park, Y-S Jeong, Visualization of the constrained coverage of mobile sensor networks based on GML, in *4th International Symposium on Cloud and Convergence Computing*, Vancouver, Canada, pp. 603–608 (August 2009)
6. MA Goodrich, Potential fields tutorial. http://www.ee.byu.edu/ugrad/srprojects/robotsoccer/papers/goodrich_potential_fields.pdf
7. A Howard, MJ Mataric, GS Sukhatme, Mobile sensor network deployment using potential fields: a distributed, scalable solution to the area coverage problem, in *6th International Symposium on Distributed Autonomous Robotics Systems (DARS02)* (June 2002)
8. S Poduri, GS Sukhatme, Constrained coverage for mobile sensor networks, in *IEEE International Conference on Robotics and Automation*, New Orleans, LA, pp. 165–172 (2004)
9. J Chen, MB Salim, M Matsumoto, A single mobile target tracking in Voronoi-based clustered wireless sensor network. J Inf Process Syst. **6**(4), 17–28 (2010)
10. G Wang, G Cao, TL Porta, Movement-assisted sensor deployment. IEEE INFOCOM 2004. **4**, 2469–2479 (March 2004)
11. OpenGIS Consortium, Inc, Geography Markup Language [GML], 07-036_Geography_Markup_Language_GML_V3.2.1.pdf.
12. OpenGIS Location Service Core Serviceshttp://www.opengeospatial.org/
13. Z Guo, S Zhou, Z Xu, A Zhou, G2ST: a novel method to transform GML to SVG, in *Proceedings of the 11th ACM international symposium on Advances in*

geographic information systems, OpenGIS Consortium, Inc, pp. 161–168 (November 2003) http://www.opengeospatial.org/docs/02-023r4.pdf. Geography Markup Language (GML) Implementation Specification

14. S Shekhar, RR Vatsavai, N Sahay, TE Burk, S Lime, WMS and GML based interoperable web mapping system, in *GIS: Geographic Information Systems*, pp. 106–111 (2001)
15. M Cardei, MT Thai, Y Li, W Wu, Energy-efficient target coverage in wireless sensor networks, in *Proc IEEE INFOCOM'*. **3**, 1976–1984 (2005)
16. KK Chintalapudi, A Dhariwal, R Govindan, GS Sukhatme, Ad-hoc localization using ranging and sectoring, in *IEEE Info-communications* (March 2004)

doi:10.1186/1687-1499-2012-95
Cite this article as: Jeong *et al.*: MSNS: mobile sensor network simulator for area coverage and obstacle avoidance based on GML. *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:95.