A Swarm-based algorithm for Optimal Spatial Coverage of an Unknown Region

Marcel Antal, Ionut Tamas, Tudor Cioara, Ionut Anghel, Ioan Salomie
Computer Science Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
{marcel.antal, ionut.tamas, tudor.cioara, ionut.anghel, ioan.salomie}@cs.utcluj.ro

***Abstract* — This paper presents an algorithm for optimal spatial coverage of an unknown region by a swarm of agents. The algorithm is based on the Ant Colony Optimization heuristic which is mapped and adapted to solve the current optimization problem. Each agent will leave a virtual pheromone trail during its movement through the unknown region, either attractor or repellent, represented by a positive or negative value, that decays in time. A novel stimergy technique is used to coordinate the agents' behavior when deciding to follow or to move away from the pheromone trail, depending on the pheromone value. For exploring the unknown areas a combination of a greedy technique based on the concept of rejection vector and a probabilistic technique for selecting the agents' rotating angles are employed. The obtained results are promising showing that our solution manages to obtain a coverage with up to 40% higher than classic rejection algorithm and with up to 25% that the distributed rejection algorithm.**

Keywords—optimal spatial coverage; swarm based optimization; stimergy coordination; rejection vector.

I. Introduction

The optimal coverage of an unknown environment using a set of intelligent agents and considering different constraints is an important problem in domains such as nuclear decontamination, victim localization in indoor or outdoor environments after cataclysms (earthquakes, arson, floods, etc.), surveillance for offices or manufacturing facilities or agricultural land coverage for reducing fuel consumption.

Being an NP-hard problem, the optimal coverage of an unexplored region cannot be solved using conventional exhaustive approaches, but rather using approximate approaches which are more practical due to their ability to provide the optimal or near optimal solutions in a short time and without processing the entire search space [1].

Biology offers many clues and heuristics which can be easily adapted for solving such optimization problems. The birds and insects are employing self-organizing behavioral strategies, at the swarm level, which help them to find food, live and survive in the harshest conditions. An example, in this sense is the foraging behavior of birds in nature which was used for designing the Particle Swarm Optimization heuristic [2]. When searching for food, the birds of a flock cooperate with each other to find a rich food sources without being centrally coordinated. The flock may identify an area rich in food if each bird's trajectory is established by efficiently combining the information regarding the bird's current position and former position where it has found a rich food area on one hand, and with data regarding the position of a neighbor bird that emits the loudest sound on the other hand [3].

The study of bio-inspired, swarm algorithms and their vast number of applications has drawn the attention of researchers from various study domains. The main benefits of using swarm heuristics over classical, centralized approaches can be clearly seen when the solution space of the problem is highly dynamic, unpredictable, unstructured and no prior information on the environment exists. These characteristics are present in case of the optimal coverage of an unknown environment using a set of intelligent agents (swarm of agents). A distributed, decentralized approach for guiding the swarm of agents to reach a common goal or to solve certain tasks as a group provides robustness, scalability and improved fault-tolerance over centralized techniques.

In this paper, by inspiring from the biological behavioral strategies of swarms which have ensured the survivability of different biological species over time, we describe an algorithm for optimal coverage of a region by swarm of agents. The agents perform the scanning of the region locally, considering real-world constraints such as coverage and communication area for each agent, failure probability, etc. Each agent has a limited scanning range through which it can have a 360 degree view with a certain radius, a view similar to infrared sensors placed on mobile robots. Also, the agents can communicate within a certain communication range and each agent is susceptible to transient or permanent failure. The main objective of our approach is to maximize the total covered area by the swarm of agents and to reduce coverage redundancy in an unknown region.

Our algorithm is based on the Ant Colony Optimization heuristic [4] which is a technique for solving optimization problems that can be abstracted and specialized for finding optimal paths through graphs. Each agent will produce a virtual pheromone during its movement through the unknown region, either attractor or repellent, represented by a positive or negative value, that decays in time. The pheromones influence the behavior of other agents which decide to follow or to move away from the pheromone trail, depending on the pheromone

978-1-4799-1494-4/13/$31.00 ©2013 IEEE

value. If no pheromones are found, the agent will move in the direction of the largest uncovered area of the current location.

The major contributions throughout this paper are the following:

- A novel stimergy technique for agents' coordination that uses two types of pheromones, repellent and attractor, in order to take advantage of the current coverage information, to reduce redundancy and to increase area coverage.
- A combination between a greedy and probabilistic technique for individual exploitation of an unknown area while taking into consideration the physical distribution of the swarm, which gives a good heuristic at agent level when dealing with complete unexplored region.
- Extensive experimental simulations for comparing our solution to state of the art coverage algorithms when varying different parameters. The obtained results show the effectiveness of our algorithm.

The rest of the paper is structured as follows: Section 2 presents relevant background literature; Section 3 describes our swarm-based algorithm for optimal spatial coverage of an unknown region; Section 4 shows comparative results proving the effectiveness of our solution compared with state of the art ones, while Section 5 concludes the paper.

II. Related Work

The current section presents a critical analysis of relevant state of the art solutions for optimal coverage of an unknown region given a set of agents.

In [5] a coverage algorithm is described which considers pixel-based grained environment with local communication and limited memory, where robots use a wall-following algorithm, grouped as a chain directed by a leader. The leader of the formation tries to detect a frontier form the neighboring cells and in case of failure, it applies breath-first or depth-first algorithms for frontier detection. This approach does not take into consideration the problems that appear when the connectivity of the chain of agents is lost. The costs needed to keep the chain of agents connected increases the number of steps needed to cover the environment, as opposed to our algorithm, where the loss of connectivity in the swarm does not significantly influence the overall performance.

In [6] mobile robots that can detect obstacles and other robots through a line of sight, but without any real point-to-point communication mechanisms are considered. The behavior is determined by a combination of simple coverage heuristics such as random walking, following a wall in the environment, exploring open spaces for fast initial dispersion, etc. A *fiducial* technique which uses wireless signal intensity from each robot in order to keep the swarm dispersed is proposed in [7]. This technique doesn't take into consideration the problem of coverage redundancy. Ludwig et al. [8] extends this work by adding wireless signal intensity as a rough approximation of distance between robots. The Clique Intensity Algorithm described considers each robot as a node, finds maximal cliques and disperses the swarm based on clique intensity.

In [9], a coverage algorithm based on a combination of physical forces with a probabilistic behavior for each agent-robot is presented. Attraction to open spaces and rejection to obstacles are modeled through physical forces. Loss of connectivity is minimized by broadcasting signals to neighbors and decisions at each step are taken through a probabilistic approach. A similar technique is also used in our algorithm; however, in our approach, the agents are indirectly influencing each other through the virtual pheromone concept making them more resilient to failures due to the loss of connectivity.

Techniques where some knowledge of the environment exists a priori are also proposed in the state of the art literature. In [10] the terrain is divided in cells using weights based on difficulty of traversal and the region is covered using tree traversal algorithms. In [11] a similar decomposition is performed and a minimum spanning tree-based algorithm is used for optimal traversal between the region cells. The major difference between this technique and our algorithm is the use of a priori knowledge of the region which makes the problem simpler and deterministic algorithms can be employed for optimal coverage.

A thorough and detailed presentation of conventional and swarm-based coverage algorithms is presented in [12]. The authors define for each robot a communication range, in which the robots can communicate and exchange information, a coverage range, in which the robot can detect the environment and memory constraints, to emulate more accurately the actual agent executing the algorithm in real world and to prevent from keeping a complete map of the entire environment. At each step, a robot acts depending on the robots that are present in its communication range or in its coverage range as follows: (i) if other robots are present within its communication range, the robot acts to maximize the distances from them by using a probabilistic dispersing algorithm, and (ii) if robots are present in its coverage range, the robot will employ a node counting algorithm for determining the virtual pheromones of the neighboring positions and computes the information gain associated with each action defined as the information entropy for the robots within the communication range. A probabilistic algorithm is used for selecting an action based on this information entropy. We have also modeled the agents as having a communication and coverage range and the capability of leaving a virtual attractor pheromone in the environment; however, we have extended this technique, in our case the agents being able to leave repellent pheromones in the environment.

Starting from the disadvantages of classical coverage algorithms, the distributed, bio-inspired, probabilistic technique that we have developed brings a number of clear advantages over classical, deterministic approaches. The *fault-tolerant nature* of the algorithm is achieved by the ant colony optimization heuristic (i.e. our algorithm is able to cope with transient or permanent failures of the agents). *Redundancy of the scanned regions is minimized* using virtual pheromones and by employing a greedy exploration technique of the unknown region. Our proposed solution is able to perform well even for large number of unknown environments types, where the density or placement of the obstacles varies. Finally, our algorithm *robustly manages* a number of conditions that occur often in the real world that influence the agent itself, such as the sensorial perception or communication. The performance of our solution increases linearly with the communication and coverage range of the agents.

III. Swarm-based Optimal Area Coverage Algorithm

We define the region that needs to be covered as a continuous, 2D region that can have any number of obstacles and at least on point of entry. In our approach an agent ag emulates the behavior of a physical robot that is able to move and collect information from the region. An agent can communicate within a certain communication range, $ComR_{ag}$, and can scan the region from their current position within a certain coverage range, $CovR_{ag}$.

An *exit location* is a location situated on the circle with the radius equal to the agent's coverage range, to which the agent can physically move from its current location.

The *Coverage Area*, $CovA_{ag}$, of a certain agent ag at a given position (x_{ag}, y_{ag}) is defined as the set of points from the region that are within the disk centered at (x_{ag}, y_{ag}) and with the radius equal to the coverage range $CovR_{ag}$ of that agent:

$$CovA_{ag} = \left\{(x, y) \mid \left(x - x_{ag}\right)^2 + \left(y - y_{ag}\right)^2 \le CovR^2{}_{ag}\right\} \tag{1}$$

In a similar manner we define the *Communication Area* $ComA_{ag}$, of an agent ag as:

$$ComA_{ag} = \left\{(x, y) \mid \left(x - x_{ag}\right)^2 + \left(y - y_{ag}\right)^2 \le ComR^2{}_{ag}\right\} \tag{2}$$

where $ComR_{ag}$ is the communication range of that agent.

The main goal of our swarm-based optimal area coverage algorithm is to obtain a maximum coverage of the region over minimum time steps, while robustly handling transient and permanent failures of the agents and minimizing redundancy of covered area by using more than one agent. The algorithm tries to keep the swarm connected, meaning that any two agents can communicate either directly or by intermediate agents. Consequently, each agent has at least one other agent at a distance less or equal to the communication range. As an effect, the agents should have the same environment maps and pheromone maps at a given time because the information exchange can occur between any two agents.

The pseudo code of the algorithm is presented in Fig. 1. Starting from its current position, the agent determines the positions of the neighbor agents located within its communication area and the available exit locations within its coverage area (lines 2, 3). While determining its neighbors, the agent also exchanges information with them. The local environment map and pheromone map are sent to the neighbors. Each neighbor sends back its own map. The agent merges its local maps with the maps received from the neighbors. The maps are merged with the rules:

i. Environment map: $merged_{map[i][j]} = max(map_1[i][j], map_2[i][j])$;

This means that with the encoding presented above, the highest priority is given to obstacles, then free space and finally unknown area. Even if in an optimal case, the local maps are identical in their intersection areas, due to noise and errors they might differ. Using the rule presented, the errors are somehow eliminated. A more precise approach would be to give some credibility to each agent and choose the new element by extracting from a probability distribution generated from this credibility. Consequently, the agent with higher credibility will fill in the map more often.

ii. Pheromone map: $merged_{map[i][j]} = max(map_1[i][j], map_2[i][j])$;

However, problems might appear when two agents are exchanging maps at consecutive time steps, when almost no changes are made, having as effect doubling the pheromone values. Some communication protocols should be established.

If the agent is in an unexplored region (i.e. a location that it was not visited by any other agent), the decision process to select the next exit location in the region to which it will be headed is taken by *Individual Exploration* based on computing a *rejection vector* (see line 6 and sub-section III.B). Taking into account the number of valid exit locations detected on its coverage area, the agent will behave as follows:

i. If the agent doesn't find a valid exit location or it reaches an obstacle then it will go back to the previous location and pick the new direction by probabilistically determining the rotation angle using a Gaussian distribution (lines 8-10);

ii. If the agent finds exactly one valid exit location it will choose this location and will place repellent pheromones on its current position to indicate that there is no need for other agents to come and explore further (lines 11-14);

iii. If the agent finds more than one valid exit, it will always choose the one which has the largest unknown area (lines 15-19).

If the agent is located in a position where it has some prior information in form of pheromone trails left by other agents that have been through this location (line 23), it will select the next exit location based on a *Stimergy Coordination technique*. The moving direction is selected as the one with the highest level of pheromones (i.e. gradient based pheromone trail following on line 23) and if there are any valid exits, it will choose the one with the largest uncovered area (lines 25, 26) and it will decrease the level of pheromones in the connected area (line 27) by placing repellent pheromones. The uncovered area is referring to the area at an exit location that was not already visited by other agents. In the case that no valid exits are available, the agent will perform a probabilistic backtracking technique (line 29).

Algorithm 1 SWARM-BASED OPTIMAL AREA COVERAGE

Inputs: Agent's current location

Outputs: Agent's movement to the next exit location

```
1.  Begin
2.    neighbAgents = DetermineNeighborAgentsPosition();
3.    noValidExits = DetermineValidExitsFromCurrentPosition();
4.    if UNEXPLORED_AREA then
5.      // Individual Exploration
6.      ComputeRejectionVector(neighbAgents);
7.      switch noValidExits
8.        case 0:
9.          GoBackAndRotate();
10.         Break
11.       case 1:
12.         MoveToIdentifiedExitLocation();
13.         PlaceRepellentPheromones();
14.         break
15.       default:
16.         ChooseExitLocationWithLargestUncoveredArea();
17.         MoveToSelectedExitLocation();
18.         PlaceAttractionPheromones();
19.         break
20.     end switch
21.   else
22.     // Stimergy coordination
23.     FollowPheromoneGradientTrail();
24.       if noValidExits >0 then
25.         ChooseExitLocationWithLargestUncoveredArea();
26.         MoveToSelectedExitLocation();
27.         PlaceRepellentPheromones();
28.       else
29.         GoBackAndRotate();
30.       end if
31.   end if
32. end
```

Fig 1. The swarm-based optimal area coverage algorithm

A. *Stimergy Coordination*

The *stimergy coordination* among the agents is achieved by means of pheromones. We consider the pheromone a substance left by each agent on the surface of o a location when it passes through that location. The pheromone left behind by an agent can be detected by any other agent that visits that location. The agents do not keep any type of partial or global environment map (locations that have been already been visited) or pheromone map (the value of pheromones at visited locations).

We define the value of the pheromone at the location (x, y) for the time t as $Pheromone(x; y; t)$. We have considered two types of pheromones:

i. Attractor pheromone: defined by a positive value and indicates that the agent should continue following this trail;

ii. Repellent pheromone: defined by a negative value and indicates that the agent should take another route further, away from this location.

The reason for defining and using two types of pheromones can be easily seen in the example from Figure 2. Let us consider a simple Y -like corridor that must be explored by an agent. At first the yellow agent goes on an unexplored region and until it reaches the bifurcation. Then it will place repellent pheromones (-), on the bifurcation path it is following since it is currently exploring this region and there is no need for other agents to do so. In this way the unexplored region are marked for stimulating other agents to visit them.

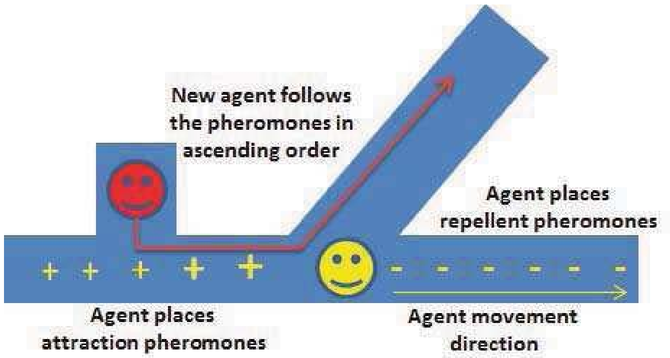

Fig. 2. Example of using attractor and repellent pheromones

To achieve the *stimergy coordination* among the swarm of agents, the explored regions are marked with pheromones in a gradient based manner (i.e. different intensity for the pheromone values). The following rules are used to update the pheromone values:

i. $Pheromone(x; y; t+1) = -1$, repellent pheromone placed in the scanned area, as long as only one exit is encountered;

ii. $Pheromone(x; y; t+1) = Pheromone(x; y; t) + n$, attractor pheromone is placed if n new paths are discovered;

iii. $Pheromone(x; y; t+1) = Pheromone(x; y; t) + D \times \varepsilon$, pheromone value is updated based on the distance from the agent that has placed the pheromone D, multiplied with a small, negative real value close to 0 (ε). Consequently, the pheromones are placed in a descending order from the agent's position. This way, the explored regions are marked with pheromones in a Gradient manner. When another agent decides to follow the pheromone trail, it will head always for higher pheromone values, finally reaching the source point of the pheromone trail.

B. *Individual Exploration*

In our approach the individual exploration of an unexplored region is based on a combination between a greedy and probabilistic technique, while taking into consideration the physical distribution of the swarm. The technique is based on the dispersion technique presented in [12] which we have enhanced with a probabilistic technique for agent backtracking when reaching a dead-end and with different decision criteria.

When an agent is in a yet unexplored region and no stimergy information is found, it will decide on the next location in the region to which it will be headed by constructing a rejection vector. The rejection vector contains on each position the rejection value calculated between the agent located in the unexplored region and one agent from the set of agents located in its communication range (see relation 3).

$$RjVector(ag) = \{RjValue(ag, ag_i), i:1...n\} \quad (3)$$
$$where\ ag_i\ locatedIn\ ComR_{ag}$$

The rejection value is computed by modeling the atomic forces that appear between electrically charged particles as a rejection mechanism. Each rejection value has an impact decaying quadratically with respect to the distance d between the agents $ag1$ and $ag2$:

$$RjValue(ag1, ag2) = \frac{CR_{ag1} \times CR_{ag2}}{d^2} \quad (4)$$

The rejection vector is used by an agent when deciding to go as far away as possible from the other agents in order to spread the group (i.e. curiosity approach). The agent computes the sum of the rejection values from the vector to find the direction that will maximize the distance to each of its neighbors. Furthermore, it chooses among the possible valid exit locations detected, the one that is closest to this new direction.

When taking a decision based on rejection vector or pheromone values, if several exits are at tie, the decision is taken on the largest uncovered area criterion. This means that for each of the exit points detected, the agent computes how much of the coverage area at that exit point was not already visited by other agents. After computing these values, the agent chooses the exit with the maximum uncovered area, this way minimizing the redundancy of covered area by more than one agent.

If an agent reaches an obstacle, it will need to go back to the previous position and take another route for exploring the region. A simple technique will be always to go back one step, rotate 90^o in one direction and resume exploration in that direction. However, to assure that the agent doesn't get stuck in a local maximum of the search space repeating the same actions infinitely we have employed a probabilistic approach.

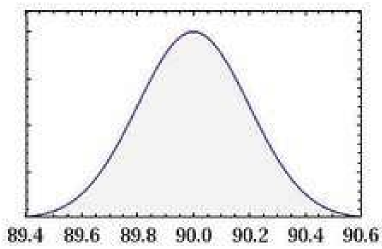

Fig 3. Gaussian distribution used for selecting the agent rotation angle

In our approach the agent will go back one step when encountering an obstacle and will rotate with an angle α that is drawn from Gaussian distribution with the mean $\mu = 90^o$ and variance $\sigma^2 = 0.2$ (see Fig. 3). In order to generate the angle α as a number with standard normal distribution, we use the Box-Muller method [13].

Each time an agent gets stuck (Fig 4), it goes back to the previous position and chooses a new direction by extracting a number from the Gaussian distribution.

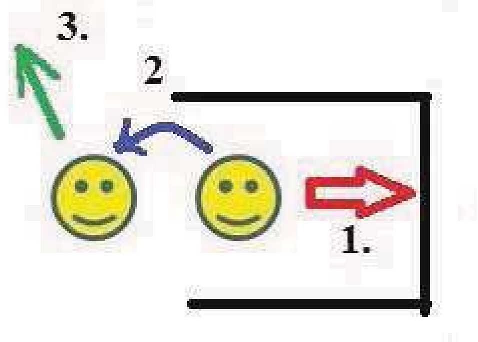

Fig 4. Example of handling the situation when the agent gets stuck.

IV. Experimental Results

The experiments were conducted in a simulated environment. We have simulated a swarm of agents, each agent modeling the characteristics and behavior of a real robot. An agent is capable of performing the following set of operations: read distance to obstacles and neighbor agents, compute its current and next location, exchange information with the neighbor agents and finally change its location by moving towards a valid exit location.

The simulation takes as input the map of an unknown region. For our experiments we have constructed three different maps presented in Fig. 4. Each map's dimension is of 900 x 900 pixels and has obstacles with various shapes that are randomly disturbed for the maps' obstacle density values of 5%, 10% and 20%. An agent places pheromones on this map by changing the intensity of the pixels in the region it explores using the pheromone placing rules detailed Section III.A.

To prove the efficiency of our algorithm, we have compared it with two similar state of the art algorithms: classic rejection algorithm and rejection vector based algorithm. The *classic rejection algorithm* is based on the characteristics of rejection forces from physics. In this case the agents will try to spread as much as possible aiming at covering as much of the unknown region as possible. The *distributed rejection algorithm* is derived from the previous algorithm. However, at each step, besides trying to spread as much as possible from each other, the agents give higher priority to unexplored regions in favor of previously visited areas.

During experiments we have tested the algorithms behavior in three different situations generated by modifying the following simulation parameters: the density of obstacles from the unknown region, number of agents composing the swarm and the communication range of agents.

In the first experiment we have evaluated and compared the three algorithms on the region maps from Fig. 5. For each map, we run one of the algorithms independently with the same simulation parameters: swarm dimension of five agents, agents' communication range larger than the map size and the agents' initial position in the left bottom corner. The simulation is run for 300 steps for each algorithm. Figure 6 results show that our algorithm has better covered the unknown region when the number of obstacles is moderate (a. and b.): coverage with up to 30-40% higher than classic rejection algorithm and with up to 20-25% than the distributed rejection algorithm. This is due to the pheromones effect of guiding the agents through large empty spaces between obstacles, thus reducing the coverage redundancy. However, if the obstacle density is higher (c.), the agents do not have to make a large number of choices when choosing a new direction, so the pheromones are not taken into consideration and our algorithm produces results similar with the distributed rejection one.

In the second experiment we have compared the algorithms behavior on the same map (Fig. 5. b - 10% obstacle density was used for tests) but this time we have varied the agents' communication range as follows: 100% of the map size, 40% of the map size and finally 20% of the map size. Figure 7 plots show that our algorithm performs better when the communication range decreases, because of the pheromone effect of guiding the agents.

In the third and final experiment we have evaluated the algorithms on the same map (Fig. 5. b - 10% obstacle density was used for tests) but this time we have varied swarm's dimension (i.e. number of agents) as follows: 3, 5 and finally 10 agents. Figure 8 plots show that our algorithm gives good results. The ratio environment size / (number of robots * coverage area) is quite large. However, in an opposite case the algorithm may perform worse than the considered state of the art algorithms because of the pheromone effect of guiding agents on the same path.

V. Conclusions and Future Work

In this paper, we have presented a swarm inspired algorithm for optimal coverage of an unknown region. Our algorithm is based on Ant Colony Optimization heuristics and brings two major contributions to the state of the art: (i) novel stimergy technique that uses two types of pheromones, repellent and attractors, in order to take advantage of the current coverage information, for reducing redundancy and increasing coverage and (ii) a combination between a greedy and probabilistic technique, while taking into consideration the physical distribution of the swarm, which gives a good heuristic at agent level when dealing with complete unexplored region.

The obtained results are promising showing that our solution manages to obtain a coverage ratio in average with up to 35% higher than classic rejection algorithm and with up to 23% that the distributed rejection algorithm. The stimergy effect of our implementation presents a clear advantage over distributed rejection approach for the case when the communication range between the robots is small compared to the environment size and the density of obstacles in the explored region is medium. However, if the agents' communication range is larger than the environment map size, the stimergy effect doesn't make a significant difference, sometimes the algorithm performing worse than the state of the art ones.

References

[1] I. Salomie, M. Vlad, V. R. Chifu, C. B. Pop, "Hybrid immune-inspired method for selecting the optimal or a near-optimal service composition", FedCSIS : 997-1003, 2011.

[2] J. Kennedy, and R. Eberhart, "Particle swarm optimization", Proceedings of the International Conference on Neural Networks, Volume 4, pp. 1942 - 1948, 1995.

[3] D. Floreano, and C. Mattiussi, "Bio-inspired artificial intelligence: theories, methods, and technologies", The MIT Press, 2008.

[4] M. Dorigo, M. Birattari, T. Stützle, "Ant colony optimization - artificial ants as a computational intelligence technique", IEEE Computational Intelligence Magazine, 2006.

[5] T. Hsiang and E. Arkin and M. Bender and S. Fekete and J. Mitchell, "Algorithms for rapidly dispersing robot swarms in unknown environments", Proc. of 5th Workshop of Algorithmic Foundation of Robotics, 2002.

[6] R. Morlok and M. Gini, "Dispersing robots in an unknown environment", Proc. 7th Int. Symp. on Distributed Autonomous Robotic Systems, Springer, Berlin, Germany, 2004.

[7] E. Ugur, A. Turgut, E. Sahin, "Dispersion of a swarm of robots based on realistic wireless intensity signals", Int. symposium on computer and information sciences, 2007.

[8] L. Ludwig and M. Gini, "Robotic swarm dispersion using wireless intensity signals", Proc. 8th Int. Symp. on Distributed Autonomous Robotic Systems, Springer, Berlin, Germany, 2006.

[9] A. Pakanati and M. Gini, "Swarm dispersion via potential fields, leader election, and counting hops", Proc. of the Second international conference on Simulation, modeling, and programming for autonomous robots, Springer, Berlin, Germany, 2010.

[10] X. Zheng, S. Jain, S. Koenig and D. Kempe, "Multi-robot forest coverage", Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IEEE Press, Piscataway, NJ, USA, 2005.

[11] N. Correll and A. Martinoli, "Robust distributed coverage using a swarm of miniature robots", Proc. Intl. Conf. on Robotics and Automation, IEEE Press, Piscataway, NJ, USA, 2007.

[12] P. Dasgupta and K. Cheng, "Distributed coverage of unknown environments using multi-robot swarms with memory and communication constraints", Technical Report No. cst-2009-1, 2009.

[13] T Shinzato, "Box Muller method", http://www.lmpt.univ-tours.fr/~nicolis/Licence_NEW/08-09/boxmuller.pdf, 2007.

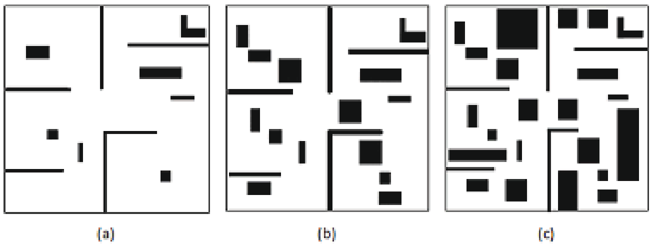

Fig. 5. Maps of unknown regions used in experiments
(a - obstacle density = 5%, b - obstacle density = 10%, c - obstacle density = 20%)

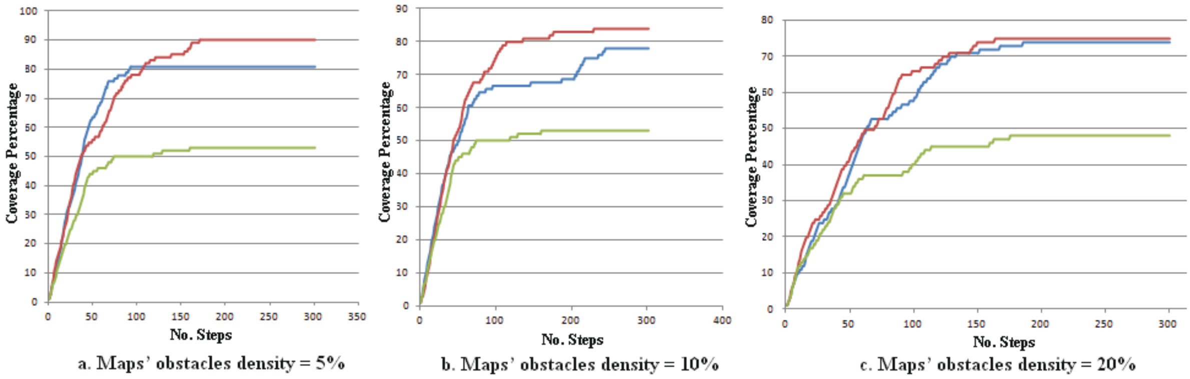

Fig. 6. Coverage area percentage results for different obstacles density values
(Red – swarm-based optimal area coverage algorithm, Blue - distributed rejection algorithm, Green - classic rejection algorithm)

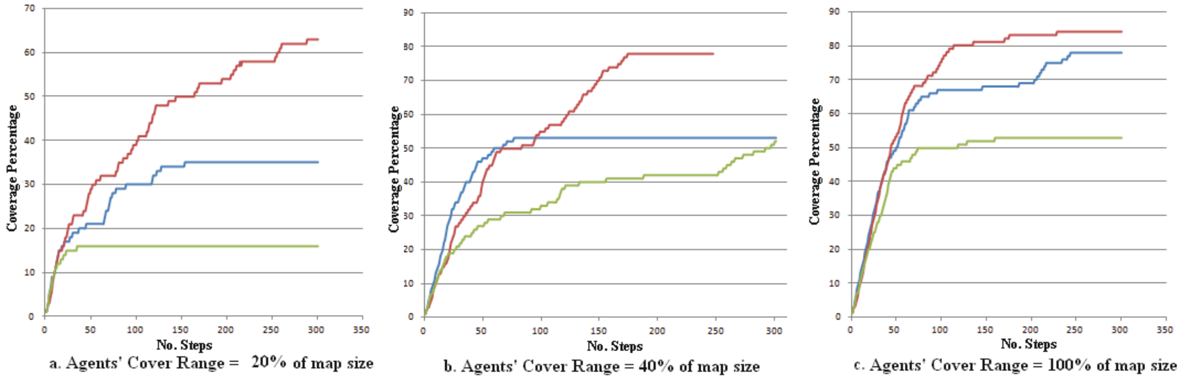

Fig. 7. Coverage area percentage results for different agents' communication ranges
(Red – swarm-based optimal area coverage algorithm, Blue - distributed rejection algorithm, Green - classic rejection algorithm)

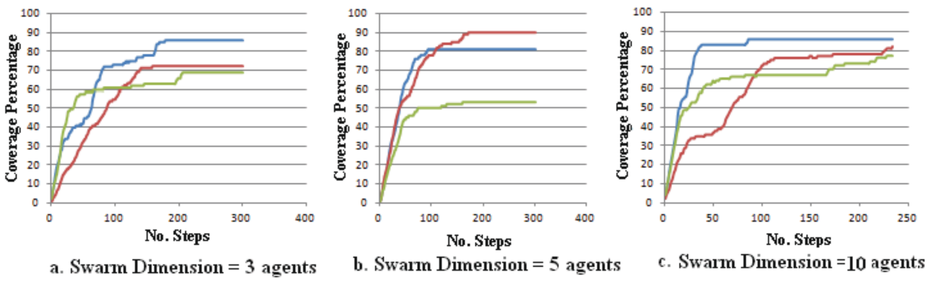

Fig. 8. Coverage area percentage results for different dimension of swarm
(Red – swarm-based optimal area coverage algorithm, Blue - distributed rejection algorithm, Green - classic rejection algorithm)