7

Multirobot Cooperative Model applied to Coverage of Unknown Regions

Eduardo Gerlein and Enrique González
Pontificia Universidad Javeriana
Colombia

1. Introduction

This chapter describes the problems of robotic exploration and coverage, and also presents the main characteristics that must include a multirobot platform in order fulfill such task in terms of general architecture, goals and cooperativeness. It is described a validated model for multirobot exploration called Backtracking Spiral Algorithm – Cooperative Multirobot, BSA-CM.

Exploration and coverage of unknown regions are tasks that have been developed by human beings throughout history. Nevertheless, tasks that imply lands under hazardous or hostile circumstances such as planetary exploration (Menzel, 2000), humanitarian demining (Freese et al., 2007), environmental cleaning (Tilden, 1996) of dangerous materials as radioactive or military remainders leads to delegating such exercise to robots. The main issue is to protect human life at all cost. In this particular case, by unloading the people of tasks in daily routine that implies covering of regions.

Exploration of unknown areas and lands, and also execution of complex tasks in unknown or partially structured regions, is an extremely complex job for an independent robot, given the nature of the algorithms and intelligence and reasoning levels whereupon are equipped the robots (Murphy, 2000). Nevertheless, the possibility of having robotic platforms able to explore these spaces is becoming more and more a fundamental necessity in the development of autonomous mobile robots. The subject of covering and exploration with robotic entities has been boarded from different optical but in case of multi-robot systems, the challenge just begins.

Systems that involve several robots can present advantages over those that include a single robot of great power, processing capacity and cost. First of all, a robot team can cover an area more quickly than a single robot. On the other hand, the exploration performed by a set of robots is robust due to the added redundancies and allows to have a notion of the best next point of view to gain information of the area where a single system would fail (Balch & Parker, 2002).

After a review of different approaches to the coverage problem with robots, it is possible to conclude that such solutions are subscribed in one of two main groups depending on the movement strategy chosen to solve the problem of coverage: those that use Non-Structured trajectories, where the navigation of the robots depends on the search of the best next point of view that derives in the elimination of borders of the unknown world, or, by means of

probabilistic methods. The other family involves Structured trajectories, where a sweeping of simple regions is made by deterministic movements following zig-zag or spirals paths.
The study of multi-robot coverage solutions allows to determine the relevant characteristics in order to be considered at the time of designing a coverage system that fulfill the requirements such as reliability, robustness and efficiency and also to determine the cooperative characteristics needed to allow the robots to act as a team. The focus of this chapter is the description and tests of BSACM - Backtracking Spiral Algorithm Cooperative Multi-robot, which is an upgraded version of a single robot coverage procedure, implemented under a multi-agent cooperative systems approach as an example of a model that includes the requirements described. It is presents the multi-agent cooperative coverage with robotic entities, and the internal agent architecture designed. The algorithm is tested in an event-driven multi-agent environment, which includes multitasking, multithread, and a continuous space robot simulator. Tests and simulation results show the convenience of using robot teams for coverage tasks and multi-agent approach to coordinate their actions.
In this chapter the reader will find:

- In section 2, the reader will find a deep analysis of cooperation in multiagent systems as a basis of multirobot platforms. Based on the theory compilation it is possible to determine, discriminate and group the intrinsic characteristics that a cooperative system must achieve in order to fulfill a successfully coverage task.
- Section 3 presents a review referred to a coverage models with a single robot, and summarize some of the multirobot solutions. It is presented a comparative analysis of those approaches. As a result it is possible to determine requirements related to architecture, world modeling and in general, the algorithms used in multirobot coverage systems.
- Section 4 presents the design, implementation and tests of BSA-CM as a case of study of a system that includes the requirements described in previous subchapters and that achieve a complete coverage with a team of robots.

2. Cooperation in multiagent systems

In this particular case, the task of region coverage will have a multiagent cooperation approach. The main goal is to give a general frame for region coverage using multirobot teams in terms of requirements and indicators that must be present and evidenced in the system. This analysis allows a designer to build the architecture in terms of cooperation since the beginning of the process. This section aims to define an agent and its architecture in the context of a multiagent system; finally, a brief description of communication, task allocation, and information access leads to the identification of the requirements that have to be taken into account to design a cooperative system.

2.1 Agents

Although there is not an enough generalized definition for what an agent represents, it is possible to mention a common characteristic between the different approaches which is the capacity to do “tasks” for the user under a certain degree of discernment, given by a basic level of intelligence which allows it to interact with the environment in an autonomous way, requiring the user intervention just in certain cases. In other words, an agent is an “entity” immerse in an environment capable of perceiving and taking actions according to those perceptions. In most of the cases, an agent can modify the state of its environment.

Following this general definition, we can include human beings, animals, software, robots as being agents. From now, we can refer to an agent as an entity created and designed by humans, limiting this category to software and robots.

An agent is considered autonomous if the type of developed actions comes from the experience of interaction with its environment and not from the previous knowledge given by the designer, even though this a priori knowledge is necessary. Agents require:

- **goals**, which orient them to the main objective and provide feedback about the progress of the general task;
- **sensors**, which allow to capture data from the environment;
- **intelligence**, in order to interpret data and orient behaviors to accomplish the main goal;
- **actuators**, which allow the agents to take actions which result in the accomplishment of the task.

2.2 Agent architecture

An agent is designed under a basic structure or architecture that answers to the fashion in which the agent is organized internally. It involves, besides sensors and actuators drivers, a main program which includes certain level of intelligence in order to interpret data from the sensors and to give directions to the actuators based in the data process to adapt itself or modify the environment. Given the extensive theory about agents and multiagent systems, there are many architectures proposed, however, it has to be taken into account how the world representation is related with the new perceptions, and how they trigger decisions about future actions (Ferber, 1999).

Thus, the mapping mechanism between perceptions and actions depends entirely on the main goal or task that the agent is designed for. For example, it is possible to analyze a scheme where certain perception is related to one or a set of actions. In this case, the designer has to analyze all the possible perceptions and situations that the agent will find. If the agent is confronted to unknown or none determined situations, it can be derived in malfunction or inactivity. When different actions could apply for a situation, it is also possible to established priority mechanisms or to identify special actions for special situations.

2.3 Multiagent Systems (MAS)

A multiagent system is formed by a set of agents that coordinate and conjugate their abilities and resources to solve particular or global problems (Ferber, 1999). A MAS can be defined by the following intuitive items:

- An environment *E*.
- A set of objects *O* located in *E*.
- A set of agents A, $(A \subset O)$ which are the active objects capable of perceive, create and modify other objects and communicate with other agents.
- A set of relations $\Re$, which relate the objects.
- A set of operations *Op*, defined to allow the agents to perceive, transform and manipulate the passive objects in *O*.
- A set of universal laws, that determine the consequences of *Op* in the particular world.

In such mutiagent context, cooperation is the key to achieve a common goal by a set of agents. The goals of a cooperative agent have to be aligned with those of the team. From now, any reference to an agent is oriented to physical cooperative agents, in other words, to a software agent that controls a robotic hardware in a cooperative multiagent context.

2.4 Cooperation

Cooperation between physical agents or robots is a complex control problem that implies a high degree of information exchange between them in order to accomplish the team goal. Any approach to the problem requires a global knowledge about the environment to be able to consider all the possible states. Centralized approaches have inherent limitations (Ferber, 1999), meanwhile distributed solutions are more efficient.

Cooperation between agents implies a shared main goal and mechanisms of avoidance or managing of conflicts in order to help other agents if necessary. This concept of "benevolence" (Balch & Parker, 2002) is applicable when all agents belong to the same user or organization. There are MAS that are not necessary cooperative in where the agents are only interested in their particular tasks and do not share goals neither information, even the taken actions can generate conflicts. One can conclude that cooperative robotics is oriented in designing autonomous societies that include robots capable to recognize, identify and solve problems inside their environment and look for negotiation of tasks in such situations.

The decomposition of a global task in particular tasks depends on the agent's abilities and resources availability. However, it is possible that the sum of abilities and resources results insufficient. In some cases, particular tasks can be incompatible even if all agents are pursing the same global goal.

The MAS can be intentionally cooperative since at early stages of design it is included this characteristic in the programming or cooperative behaviors can merge during its operation described by an external observer in a reactive way, but in both cases one can say that the system is cooperative if an *a posteriori* analysis evidence one of two conditions (Ferber, 1999):

- Adding a new member to the team increases the global performance. To accomplish this criterion, the system is in a "collaborative situation".
- Agent actions are designed to prevent or solve current or potential conflicts. To accomplish this criterion, the system is in a "conflict solving situation".

In the *intentional cooperation*, agents have the intention of cooperation at a cognitive level, and the cooperation is a mechanism to achieve concrete goals. In *reactive or merging cooperation*, a collective behavior satisfies at least one of the previous criterions; even thought there is no explicit intention of cooperation. It means that one can talk about cooperation even if the agents do not have knowledge about their environment and other agents. For this reason, cooperation implies simultaneous operation to oblation a common benefit (Jung, 1998).

2.5 Cooperation Indicators

According to Ferber, a MAS to be considered as cooperative it must exhibit some important characteristics: grouping, multiplication, specialization, collaboration, communications and conflict resolution.

- Grouping implies and homogeneous array of agents along the space.
- Multiplication brings an increment in the number of agents that will be reflected in a performance and reliability improvement. Multiplication will imply also the emergence of conflicts for physical space.

- Specialization refers to the adaptation degree of agents to perform their tasks. It can be a gradual process if the system includes learning mechanisms.
- Collaboration is determined by the strategy used by the system to assign and distribute tasks. Task allocation can be determined by centralized methods, where a central agent assign duties after a corresponding analysis (i.e. contract nets with one trader and several bidders) or by means of a "yellow pages" system which eliminates the hierarchy relations and tasks are located in a common board. Also there is a distributed approach which eliminated the concept of a central agent as in acquaintance nets or distributed contract nets.
- Communications are fundamental in a cooperative system, and can be seen as an extra perception capability trough the agents are able to exchange information, request actions and interpret the world in a global way.
- Conflict resolution is close related with collaboration since the tasks distribution mechanism must include negotiation procedures. Conflicts can emerge for physical space, resources, communication channel, access to data and information, etc.

Coordination of actions, supported in communication systems, avoid conflicts for access to resources caused by grouping and multiplication of agents. In multiagent systems emerge relations and dependencies that determine actions of the agent. Resources will be always limited, thus agents must include mechanisms that allow to share and to access to them avoiding obstructions, optimizing operating costs, and reducing redundant actions. Coordination is the set of activities subjacent to the main goal that allow the execution of particular actions in time and space, in a coherent and synchronized way, which derives in the increasing of the global performance and the avoidance of conflicts.

Rules that determine coordination between agents are determined by the environment in which the MAS operates. For example, a system for flight control in an airport, the coordination rules will be designed to a division of resources physical (tracks, hangars, access points, etc.); for a coverage robot team, coordination rules will look forward to make an efficient distribution of the agents along the surface.

2.6 Requirements for a Cooperative Multiagent System

In the previous sections, the basic concepts of agents and multiagent systems have been introduced, so now it is possible to define the requirements that a multiagent system must incorporate to be considered as cooperative. Designing a cooperation strategy will be based on them; the requirements have to be covered one by one, independent of the chosen methods to manage communications, coordination and synchronization.

A single cooperation model can be implemented for a diverse set of tasks, but in order to reveal and determine the cooperation level; those requirements must be measurable in a certain way in terms of existence or inexistence or by means of ratios and numeric values in order to be evaluated on subsequent stages of implementation on simulated and/or real platforms.

Deployment: it is referred to the spatial distribution of the agents. For the particular task of coverage, the physical space is a resource and a goal at the same time. The capability of the agents of getting closer or establish wide distributed arrays will determine the communication strategy. Some applications need the agents as close as possible, but others look for exactly the opposite situation improving the coverage efficiency. A harvest system for example will implement mechanism of grouping and deployment both synchronized in time.

Multiplication: as a cooperation indicator, multiplication is desired since adding new members to the team will increase the global performance and efficiency of the system. In physical environments, the addition of agents will imply the use of space and the merging of obstructions, so an efficient task allocation mechanisms must be implemented. Multiplication can offer some advantages depending on the system and the main task the system is designed for; in some cases it is desired to have some redundant task, but in others, it can affect directly the final efficiency.

Communications: for intentional cooperation, the designer must implement explicit and intentional communications between agents that informs their internal states and and the new acquired data of the world. At the same time, agents must be in capacity to interpret all the communicative acts in order to transform them in useful information for the achievement of the task. Communications are the base in the operation. It is possible that the system presents robustness to communication failures adding redundant tasks; however, communications must be present to consider the system as cooperative. Communication mechanisms, messages nature and conversational structures must be independent from the chosen channel and protocol.

Totipotence: it is referred to the capacity of the agents to execute a wide range of tasks; totipotence is the opposite feature to specialization. Agents that have this characteristic are in capacity to assume other's roles in case of obstructions or failures. Even though specialization will emerge in cases where agents develop certain skills to perform particular subtasks, and in systems with some level of learning mechanisms; however, a very high level of specialization will affect the system's robustness to agent failures.

Collaboration: this is one of the fundamental requirements in cooperation, since implies tasks distribution according to abilities and availability. It is necessary to implement a task distribution mechanism that increases the global efficiency while aiming to balance operation loads and avoiding agent inactivity.

Coordination: coordination mechanisms must synchronize actions of multiple agents at early stages to improve the efficiency of the collaboration techniques. It is necessary to implement global planning as a mean of organization and articulation of particular tasks.

Conflict solving: it is important to design and implement mechanisms of negotiation and arbitrage in case of merging of incompatible goals or insufficient resources. For region coverage, conflicts will be determined for physical space, exploration routes, obstructions, and communication channel interference.

Competence: cooperation levels can be increased if competitive relations are implemented between agents since the assignation of a particular task can be determined by the profit that certain agent will obtain doing it, understanding profit as benefit minus cost. In this way, resources as energy can be optimized.

Functional Architecture: agents must be in capacity of develop those tasks that the primary goal requires. At functional level, agents must incorporate abilities according to the task and environment the system is designed to operate for. Physical agents (robots) designed for region coverage must incorporate adequate locomotion mechanisms, also sensorial systems to navigate and interpret the world, and sufficient communications modules.

World representation: the system must incorporate mechanisms to interpret data obtained from sensors as a representation of the environment and also the agent's location in the space. Precision of this representation will determine future decisions. Also the system must be able to interpret the world in a global way based on particular information from the agents. More complex mechanisms of world representation can include prediction of future states of the environment and probabilistic models that aid the decision making processes.

Robustness: the global system must accomplish the main goal and react adequately in case of failures of communications or agents. However, it is necessary to establish the minimum operation conditions in which the system continues working.

Efficiency: the system must incorporate means to measure its efficiency in terms of resource wasting, time, number of agents, and balance of loads.

It is expected that the described requirements would be present and evidenced during the system operation. As was previous mentioned, some of them must be measurable in terms of its presence or absence, while others can be expressed as a quantity or a ratio. To measure the accomplishment of the requirements in a cooperation model allow to evaluate how the system is been benefited with the cooperation strategies. But even more than just to measure the final efficiency in terms of goal accomplishment, it is necessary to create measure systems that can be implemented over real or simulated platforms under controlled conditions to determine how a system take advantage of the cooperative potentials.

3. Multirobot cooperation for exploration/coverage

The goals of the exploration and coverage tasks can be quite different: construct an incremental representation of the explored region or generate a safe navigation path; however, from the structural point of view, both tasks have to assure completeness. The subject of covering and exploration with single robotic entities has been boarded from different optical but in case of multi-robot systems, the challenge just begins.

Systems that involve several robots can present advantages over those that include a single robot of great power, processing capacity and cost. First of all, a robot team can cover an area more quickly than a single robot. On the other hand, the exploration performed by a set of robots is robust because of the added redundancies and allows having a notion of the best next point of view/coverage to gain information of the area where a single system would fail (Balch & Parker, 2002).

After a revision of different approaches to coverage with robots, it is possible to conclude that such solutions are subscribed in one of two main ways to solve the problem of coverage based on the chosen movement strategy: those that choose Non-Structured trajectories (Doty & Harrison, 1993) (Pirzadeh & Snyder, 1990), where the navigation of the robots depends on the search of the best next point of view that derives in the elimination of borders of the unknown world, or, by means of probabilistic methods. The other family involves Structured trajectories (Choset & Pignon) (Zelinsky, et al., 1992) (González, et al. 1996) (Gabriely & Rimon, 2002) (Gonzalez, et al., 2005), where a sweeping of simple regions is made by deterministic movements in zig-zag or spirals paths.

The work proposed by Choset (Choset, et al., 2004), presents an extension of the covering system described in previous projects, where a parallel sweeping is made by means of boustrophedic trajectories, subdividing the team and the exploration area as the obstacles are found. This system could not guarantee complete coverage if one of the members of the team does not complete its task. The collaboration is evidenced by means of parallel navigation paths.

Simmons (Simmons, et al., 2000) presents a multi-robot exploration approach driven by the elimination of borders with a protocol based on a bidding criterion. Depending on the assignation algorithm it is possible to obtain optimal results if the plans only consider what will happen in the near future. This system requires constant communication with a central agent, otherwise the complete system will fail.

Zlot (Zlot, 2002) describes an efficient and robust distributed exploration method using a robot team derived from the model of markets or contract net, maximizing the obtained data by the agents and reducing the execution costs as collective trip distance of the system. This system is robust since the exploration is completely distributed nevertheless, although a map is constructed with telemetric sensors it does not arrive at all the reachable points of the exploration area.

Wagner (Wagner, Lindenbaum & Bruckstein, 1999) presents a coverage system that does not require explicit communication between the robots. On the contrary, the system uses volatile signs in order to mark cells already visited. It is a useful system in a region with dynamic topology; nevertheless it does not construct a final representation of the world.

Butler (Butler, Rizzi & Hollis, 2000) shows an evolution of a single-robot algorithm presented in previous works to a multi-robot system. They present zig-zag trajectories and a cell division of the region denominated "Generic Rectilinear Decomposition". The system is efficient making maps integration and making the decomposition of the exploration area. Nevertheless, it holds a high re-sweeping rate and it is necessary to include methods of conflict solving caused by obstructions.

Howard (Howard, Mataric & Sukhtme, 2002) presents a system that looks for a robot deployment over the structure or area trying to maximize the sensors coverage. The algorithm is designed to be used in searching/rescuing operations and emergency zones monitoring. Nevertheless, it is limited to the number of robots because the deployment is performed in line of view.

The study of multi-robot systems allows to determine the relevant characteristics in order to be considered at the time of designing a coverage system that fulfill the requirements such as reliability, robustness and efficiency.

3.1 MAS coverage characteristics

When analyzing the coverage systems described below, it is possible to identify common characteristics. Those criteria must be included in a MAS designed for the execution of a surface covering algorithm.

In order to design a cooperative model for a coverage MAS, it is necessary to clearly establish the parameters that, in last instance, become the requirements for an ideal algorithm. Those criteria are:

Task Sensor Requirements: the task the system is designed for, determines the sensorial equipment to be embedded in the robots. Telemetric and contact sensors seem to be indispensable, however in an exploration task more complex and powerful sensors must be integrated, for example long range lasers or artificial vision systems.

Extension from a Single-Robot System: it is possible to use an algorithm validated on single-robot platforms in order to design a cooperative surfaces sweeping algorithm and adding the multi-agent perspective. Some of the most efficient approaches comes from single-robot solutions. The sweeping strategy must be robust to shape, size and configuration of the environment and the obstacles. The good features of a single robot algorithm are usually inherited by the multirobot one.

World Representation: it is indispensable that the system implements a world representation model used as tool for task planning; this representation is also going to be

one of the final result of the coverage procedure. One of the main goals of a coverage algorithm is to create global maps that represent the sweeping area. It is necessary also to determine the a priori information that the system must incorporate previous to the algorithm execution (i.e. shape and size of the environment and obstacles, self-location, etc.).
Trajectories: they can be planned (i.e. zig-zag or spirals) or random (systematic border elimination). Random trajectories present a high re-sweeping rate although the procedure results simple and fast. Planned structured trajectories require higher processing capacity but decrease the re-sweeping rates. Zig-Zag or spiral like trajectories implement systematic and deterministic navigation procedures, they overcome in efficiency random trajectories but imply complex design and a reliable localization system. The indicator of trajectories quality is related with the re-sweeping rate expressed as the percentage of the accessible surface that is visited more than once.
Operation Time: a general objective of a cooperative multi-agent system is based in the fact that a team of agents must reduce the operation time for the task compared with a single entity.
Completeness: it is mandatory that the system shows a complete coverage of 100% of the area, which implies to visit to all the reachable points of the region.
Competition: with the implementation of competitive relations between the team members. It is possible to increase the cooperation level since the assignment of a particular task is assigned to the agent that can get a better profit.
Architecture: at functional level, the agents must own such abilities according with the primary objective, the task and the topology of the area. In particular, physical agents must have adequate means of locomotion, sensorial systems in order to perform navigation and interpretation of the world. Also, communication systems are needed to fulfill the requirements previously mentioned.
Robustness: the system must be robust fulfilling the task and maintaining the operation even if there are communications or agents failures.
A Priori Knowledge: it is desirable that the system does not have any priori knowledge of the world, which means that the sweeping algorithm must be robust to the geometry of the accessible surface.
Initial Condition: before starting the task, the system must assume certain minimal operational conditions. When structured paths are used, usually some certain conditions as the knowledge of the initial position of the other team members are required.
End Condition: there must be a set of well-defined end conditions of the algorithm in order to determine the culmination and the success of the task. End conditions usually are associated to the covering of all the borders of the non-visited areas or the lack of reachable unknown points.

4. BSA coverage algorithm

The Backtracking Spiral Algorithm was first proposed for a single robot approach (Gonzalez, et al., 2005). This algorithm decomposed the accessible surface in regions that can be covered by structured spiral paths. Once the basic BSA algorithm is introduced, in this chapter, its multirobot extension is presented.

4.1 Single Robot BSA Algorithm

The basic BSA algorithm introduced in (Gonzalez, et al., 2005). assures the complete coverage of non-occupied cells. The map is represented by a coarse-grain occupancy grid, where cells are of the size of the robot. BSA uses two main concepts: covering of regions using a structured spiral-like path and linking of these regions using a backtracking mechanism. The model of the accessible surface is constructed incrementally as the robot navigates. Initially, all cells are marked as "unkown". When a cell is covered by the robot, it is marked as a "virtual obstacle"; these ones are not accessible to the robot while executing a spiral path. Cells with obstacles, even partially occupied ones, are marked as a "real obstacle".

Spiral structured paths are formed by concentric rings that generate a continuous path from the region's boundary (nearby obstacles) to a central spiral ending point. Before starting a spiral path the robot is placed nearby an obstacle (real or virtual), which is located at its Reference Lateral Side (RLS); RLS indicates the relative direction where obstacles have to be referenced during the spiral filling procedure. The Opposite Lateral Side (OLS) identifies the antipode of the RLS. The following set of reactive rules drive the robot to generate a spiral coverage path:

```
RS1 IF (obstacles_all_around)
       THEN ending_spiral_point_detected
RS2 IF (NOT obstacle_in_RLS)
       THEN turn_to(RLS) and move_forward
RS3 IF (obstacle_in_front) THEN turn_to(OLS)
RS4 OTHERWISE move_forward
```

Remark that the cells already marked as virtual obstacles are considered as real obstacles when evaluating the rules. At the end of the algorithm, the virtual obstacles will represent non-occupied covered regions. A backtracking mechanism is used to return to areas that have not been visited, where a new spiral procedure is performed. Backtracking points (BP) are detected and stored during the execution of a normal spiral path: a BP is a cell that could be the starting point of a future alternative path.

At the end of a spiral path, the robot must go to the nearest BP to start a new sweeping procedure. The robot builds the shortest path using a distance propagation algorithm; only already covered (virtual obstacles) cells are considered to generate the route from the end of the current spiral to the best BPs. During this path, the robot traverses only already visited cells; they are known to be free ones as they have already been covered. BSA finishes when there is no more possible BP, which means that there are no more uncovered surfaces.

An example of a typical BSA coverage is shown in Figure 1. Spiral paths are represented by continuous lines; notice that spirals starts as open paths nearby the obstacles that finally are "trapped", forming real spiral trajectories, in the concavities formed by obstacles. The paths used to go to the BPs are illustrated by doted lines; the BPs are always cells nearby the frontier of the already covered areas.

In the basic algorithm only the cells that are completely free of obstacles are considered as accessible. An extension of this algorithm has also been developed that also covers the partially occupied cells. In fact, these cells are covered by a wall following procedure during the first spiral ring nearby the real obstacles.

4.2 Multi robot BSA algorithm

The multi-robot coverage problem is solved using a multi-agent systems (MAS) approach; where a robot is seen as an entity that can perceive and modify its environment, and communicate with others to fulfill the goals of the system. Assuming such a cooperative and

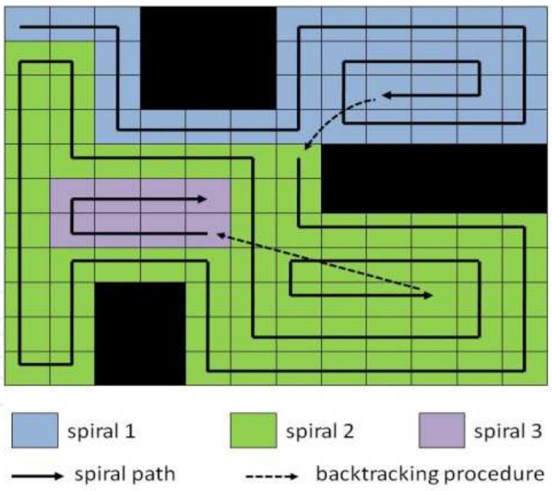


Fig. 1. Example of the basic BSA algorithm.

concurrent posture allows improving the productivity of each agent, since increasing the number of parallel tasks supposes a reduction in the total exploration time.

The BSA Conceptual framework has been extended to a cooperative multi-agent system environment in BSA-CM, where a team of robots, controlled by an agent software, are executing a sweeping procedure following the basic rules of BSA. As in the single robot algorithm, robots perform spiral paths detecting and marking backtracking points (BPs) in the map. When a robot arrives to the ending cell of a spiral path, it negotiates one of the remaining BPs. The BP that maximizes a utility function is assigned, and the robot moves to this BP and starts a new spiral path procedure.

In BSA-CM, robots must know the absolute initial position of all team members. This condition can be implemented by several methods such as: line of view, fixed positions, landmarks, etc. The robots have an identical copy replicated map; as a robot modifies a cell in its map, this information is sent to all robots; in this way, the robot team is building and sharing the same map in a cooperative way. Each robot is in one of three possible states: “inactive”, “spiral” or “return” mode; robots shall know and store the state of every team member. Normal communications are considered as “prove of life” in order to determine the activity of other robots.

A potential conflict appears while crossing from one cell to another, as several robots could try to get the same cell at the same time. A coordinated reservation mechanism is used to avoid this problem, just before traveling to the next cell, each robot broadcasts a message that includes its state, information about its current position in the map, recently discovered BPs and the next cell that is going to be reached. As in this case, all the communicative acts in the cooperative system will concern to all team members. Robots also reserve BPs once they are assigned after the negotiation process succeeds. In coverage, the main conflicting resource is the physical space, death lock situations can appear when the cell to which a robot goes is occupied or reserved by other robot. A real blocking condition will appear only in corridors or halls when both robots are in “return” mode. To solve these conflicts a role exchange procedure has been implemented.

The designed negotiation mechanism allows robots to identify the nearest backtracking points, which reduces navigation time, processing and use of resources such as energy. A robot initiates a negotiation process as soon it reach the end of a spiral path. The selection to find the best backtracking point (BBP) available, that the robot will try to negotiate, is made by means of a simulation of the estimated cost of travelling from the cell where the spiral path has finished to the candidates BPs. This simulation procedure is done only once using the current map; all the candidates BPs are ordered based on this estimated cost function and stocked in a list. Then, the bidding process starts, the goal is to determine if the cost to reach the particular BBP, the first BP on the list, is smaller than the estimated cost of any other robot to reach this BBP. If this is the case, the robot that initiates the negotiation obtains the lower cost, it will win the right to cover the BBP and the region around it. On the contrary, if the cost is greater, the bidder must select its next available best BP and starts the negotiation again. If the negotiation of the entire candidates of BP fails, the bidder robot shall select the nearest BP in order to avoid inactivity.

In order to initiate the negotiation, the bidder sent to all the other n-1 robots a message that includes the BPP and its estimated cost. Once these messages have been sent, the bidder starts a timeout counter and waits for one of these three possible situations:

a. n-1 “approval” messages have been returned (n is the number of active robots).
b. 1 o more “deny” messages arrived, which means that a smaller cost has been detected by another robot.
c. a timeout event occurs, which implies that at least one robot of the team has failed to answer in a reasonable time; in this case, it will be assumed that the backtracking point in negotiation is the best choice for the bidder robot.

During a negotiation process, the robots in “return” mode shall give automatic “approval” since its current BP assignment corresponds to the best choice available for them; in other words, a robot that is just trying to get a BP that has won during a recent negotiation process does not participate in a new auction initiated by another robot. The cost estimation for the other robots, those in “spiral” mode, is based on an optimistic assumption. First, the cost to finish the actual spiral path is evaluated by a simulation procedure that assumes all the unknown cells as non-occupied; then, the Manhattan distance from the simulated spiral end point to the BP in negotiation is calculated. The total estimated cost is the sum of these two partial costs. If this total cost is smaller than the one estimated by the bidder, the robot answers with a “deny”, otherwise it gives an “approval”.

5. Validation of BSA-CM in a simulation context

The validation of the cooperative model includes the implementation of BSA-CM in a multi-agent platform. An experimental protocol has been designed and applied in order to characterize the correctness of the extended multirobot BSA algorithm.

5.1 Implementation of BSA-CM in a multiagent platform

The system is implemented in JAVA over a multi-agent framework called BESA, which includes real time operation, multitasking, multithread and a continuous space robot simulator with uncertainly.

BESA is based in three fundamental concepts: an event-driven control approach implementing a select like mechanism, a modular behavior-oriented agent architecture, and a social-based support for cooperation between agents. The BESA architecture is composed of three levels: agent level, social level and system level. The internal architecture of an agent integrates two important features: a modular composition of behaviors and an event selector mechanism as shown in the figure 2.

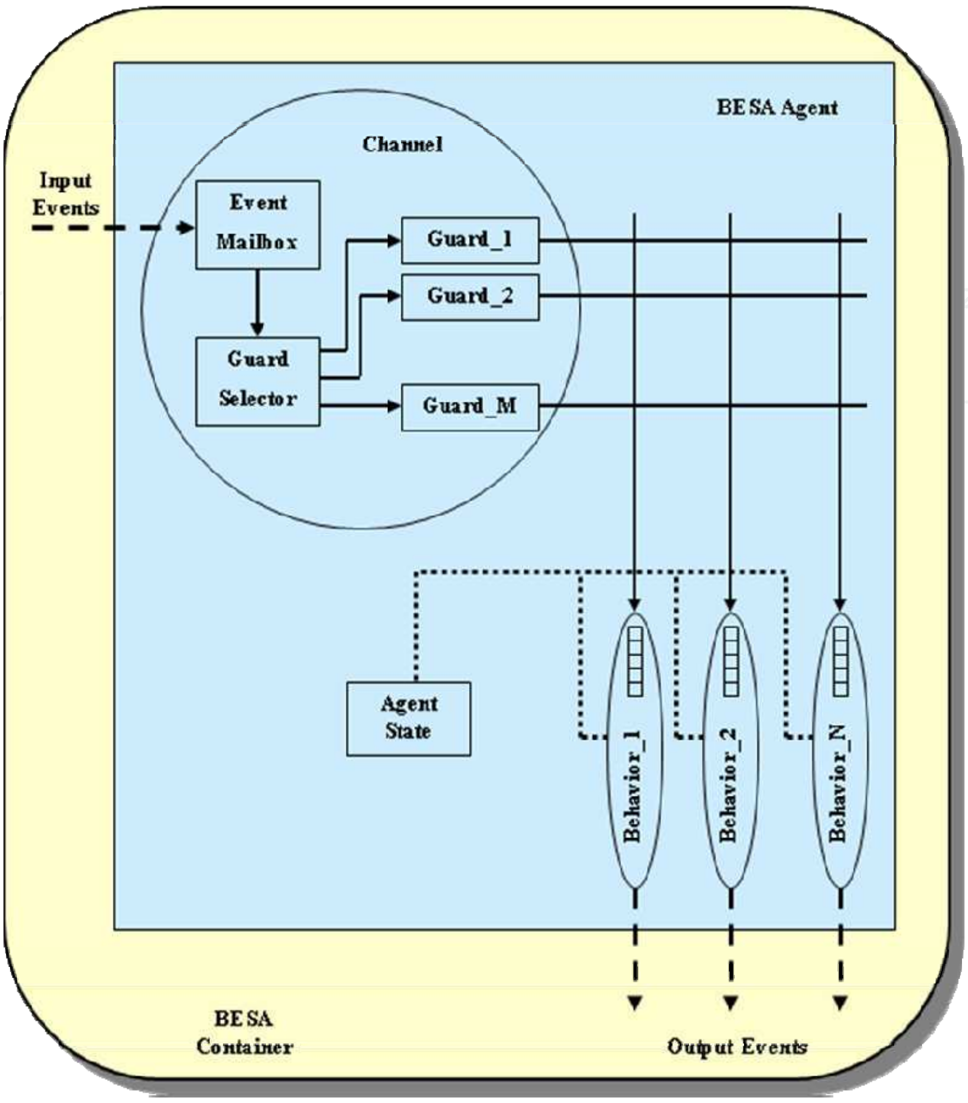


Fig. 2. General internal architecture of an agent in BESA.

The MAS for BSA-CM includes only one type of agent role, called "Explorer". In order to identify the agent internal behaviors, as well as the events associated to those behaviors, it is necessary to define the goals of the agent.

In the "spiral" mode, the robot objectives are:

a. to complete a spiral path following the simple exploration rules of BSA.
b. to identify and mark in the map: real obstacles, backtracking points (non-occupied cells in the OLS), and virtual obstacles (already covered cells).
c. to initiate a negotiation when a spiral path is finished.

In the "return" mode, the robot objectives are:

a. to select a return route through the cells previously visited.
b. to arrive to the selected BP avoiding obstacles.
c. to detect and solve corridor conflicts.

Once the particular objectives have been identified, it is possible to construct a diagram of internal modules (behaviors) and interactions of the explorer agent. This model intends to control the robot in a real time environment by executing several concurrent behaviors in a parallel way. The architecture of the AgentExplorer is shown in figure 3.

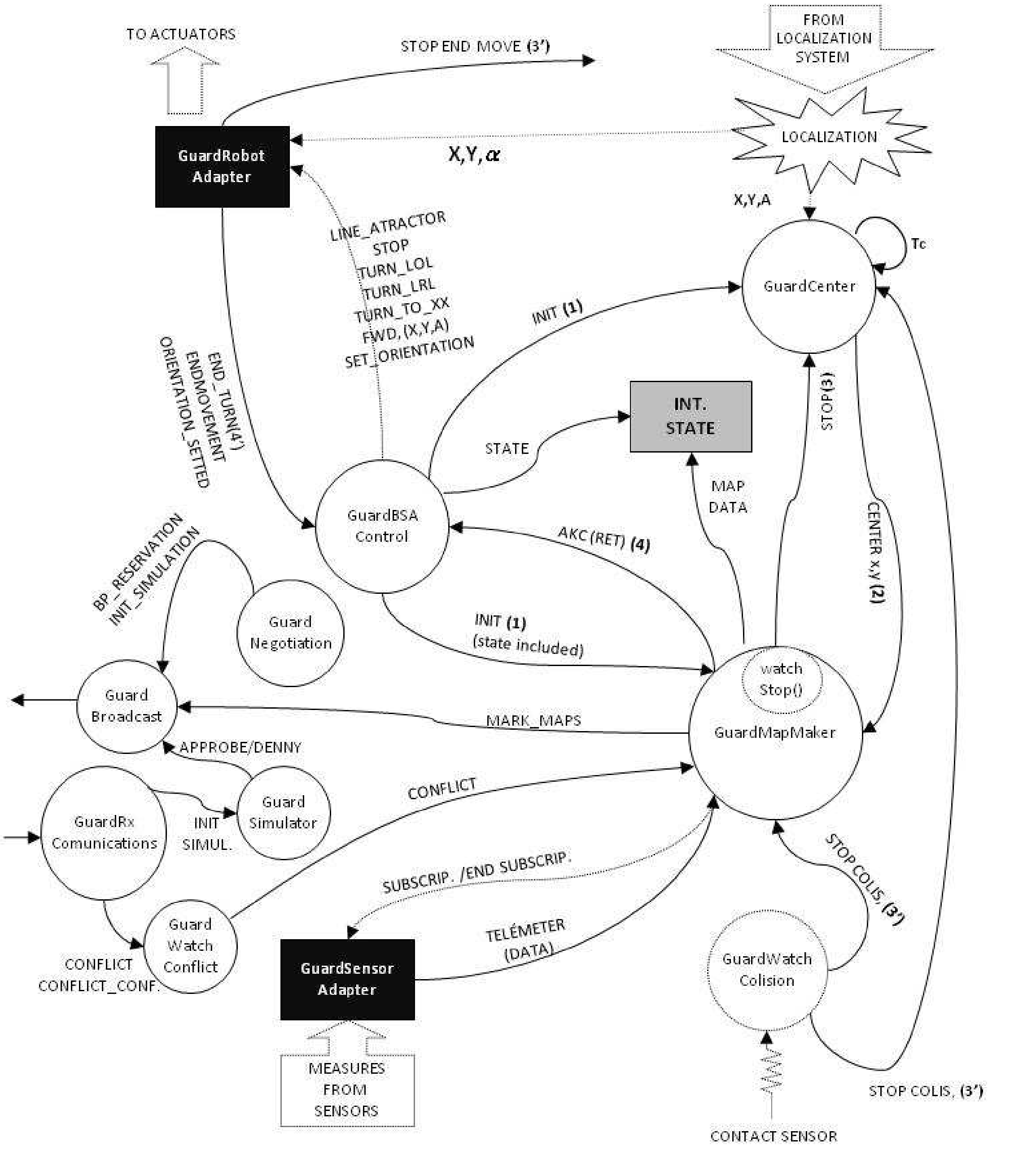
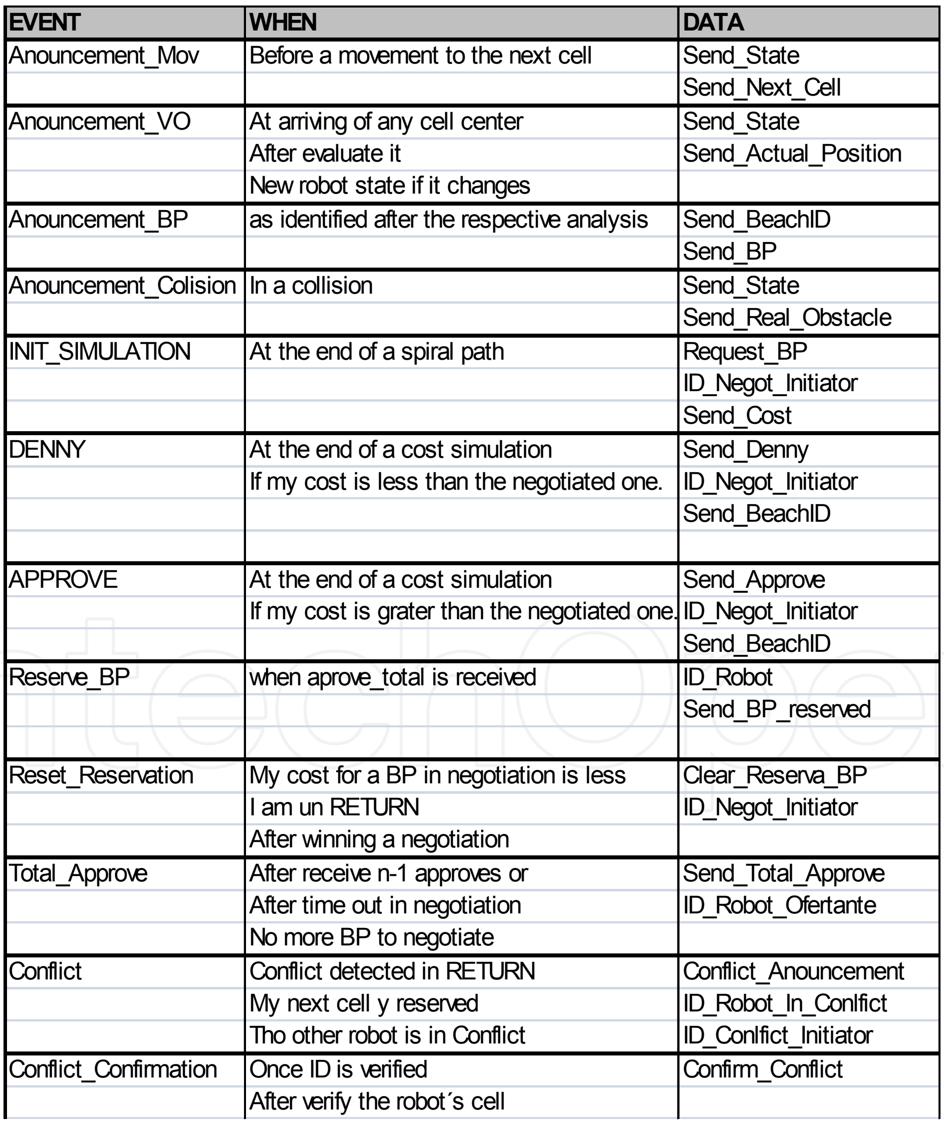


Fig. 3. Cooperative Architecture of AgentExplorer.

The internal architecture, based on a multi-agent approach, is designed to accomplish the agent objectives. It includes a main control behavior which organizes the internal interactions, sets the robot states and evaluates the BSA rules. It also includes one adapter to control the robot actuators and other adapter to acquire and manage information coming from the sensors. Additionally, each agent has several behaviors in charge of the following procedures: map marking, arriving to a cell centre detection, negotiation and simulation of spiral paths involving in the biddings, monitoring of stop conditions, detection of conflicts, and management of communications. For each defined event, a guard procedure is programmed:

GuardBSAControl: is in charge of controlling the correct execution of the algorithm.

GuardMapMaker: marks the maps by processing the sensor information and verifying stop conditions.
GuardCenter: checks the position of the agent periodically in order to broadcast the required information at every cell´s centre, before crossing to the following cell.
GuardNegotiation: negotiates the BPs and builds a return path once a backtracking point is reseved.
GuardSimulator: simulates a spiral and determines the cost for a negotiated BP.
GuardBroadcast: will send all messages concerning the team.
GuardRxCommunications: is in charge of receiving the information from other agents; it also calls the appropriate functions to treat the incoming events and information.
GuardWatchColision, GuardRobotAdapter and GuardSensor-Adapter: will supervise the contact sensors, actuators and telemetric sensors respectively.
According to the internal robot cooperation model described in the previous section, is its necessary to identify the different messages, modeled as events, exchanged by the agents. In Table 1, each event type and the attached information to it is described.



EVENT	WHEN	DATA
Anouncement_Mov	Before a movement to the next cell	Send_State
		Send_Next_Cell
Anouncement_VO	At arriving of any cell center	Send_State
	After evaluate it	Send_Actual_Position
	New robot state if it changes	
Anouncement_BP	as identified after the respective analysis	Send_BeachID
		Send_BP
Anouncement_Colision	In a collision	Send_State
		Send_Real_Obstacle
INIT_SIMULATION	At the end of a spiral path	Request_BP
		ID_Negot_Initiator
		Send_Cost
DENNY	At the end of a cost simulation	Send_Denny
	If my cost is less than the negotiated one.	ID_Negot_Initiator
		Send_BeachID
APPROVE	At the end of a cost simulation	Send_Approve
	If my cost is grater than the negotiated one.	ID_Negot_Initiator
		Send_BeachID
Reserve_BP	when aprove_total is received	ID_Robot
		Send_BP_reserved
Reset_Reservation	My cost for a BP in negotiation is less	Clear_Reserva_BP
	I am un RETURN	ID_Negot_Initiator
	After winning a negotiation	
Total_Approve	After receive n-1 approves or	Send_Total_Approve
	After time out in negotiation	ID_Robot_Ofertante
	No more BP to negotiate	
Conflict	Conflict detected in RETURN	Conflict_Anouncement
	My next cell y reserved	ID_Robot_In_Conlfict
	Tho other robot is in Conflict	ID_Conlfict_Initiator
Conflict_Confirmation	Once ID is verified	Confirm_Conflict
	After verify the robot´s cell	

Table 1. Event messages exchanged between robot agents.

5.2 Test and results

The performance of the algorithm was tested in a simulation environment using 20 “worlds” with different shape, complexity and granularity previously designed in order to test basic BSA as proposed in (Gonzalez, 2005). Each test is compared with the results in the same “world” covered by a single robot. Tests are performed also with the robots starting in two different initial positions: (a) parallel fashion - aligned close to each other and with the same heading - and (b) random distributed initial positions over the accessible surface.

The initial parallel positions can be achieved easily by real robotic platforms; this approach represents a low cost alternative to know the initial state of the robots. Figure 4 presents one of the test worlds with initial parallel positions.

Tests in each world were performed adding one robot at a time to the team until the average time of coverage is higher than the previous simulated case; in other words, the number of robots is incremented progressively until the task performance does not improve anymore.

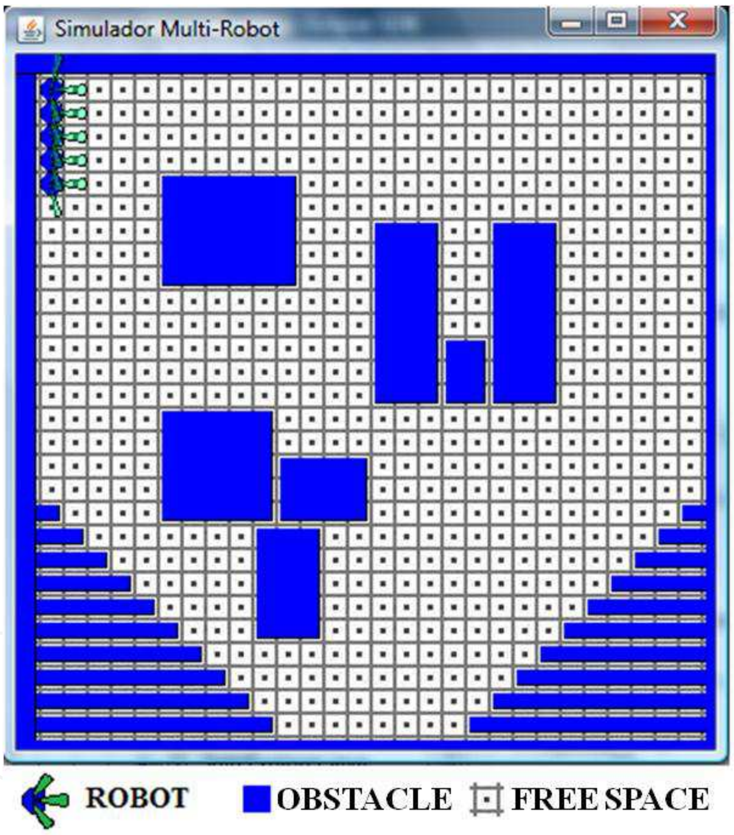


Fig. 4. Typical simulation “world” with parallel initial positions.

Total Time: is the average between all exploration times in each map. The average exploration time in all words with 1, 2, 3, 4, 5 and 6 robots is shown in figure 5. It can be noted a significant reduction of operating time with 2 robots and 3 robots. Adding members to the team reduces the time. Test with 6 robots in parallel initial positions increases the total time due to obstructions between the agents. The improvement of the performance depends on the number of robots but also on the nature and size of the world.

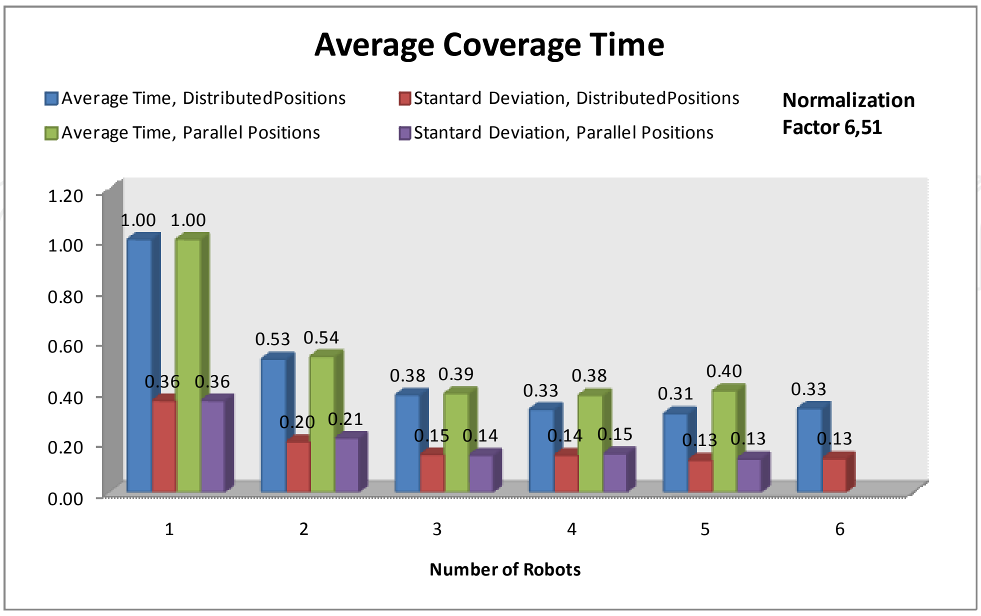


Fig. 5. Time of exploration results.

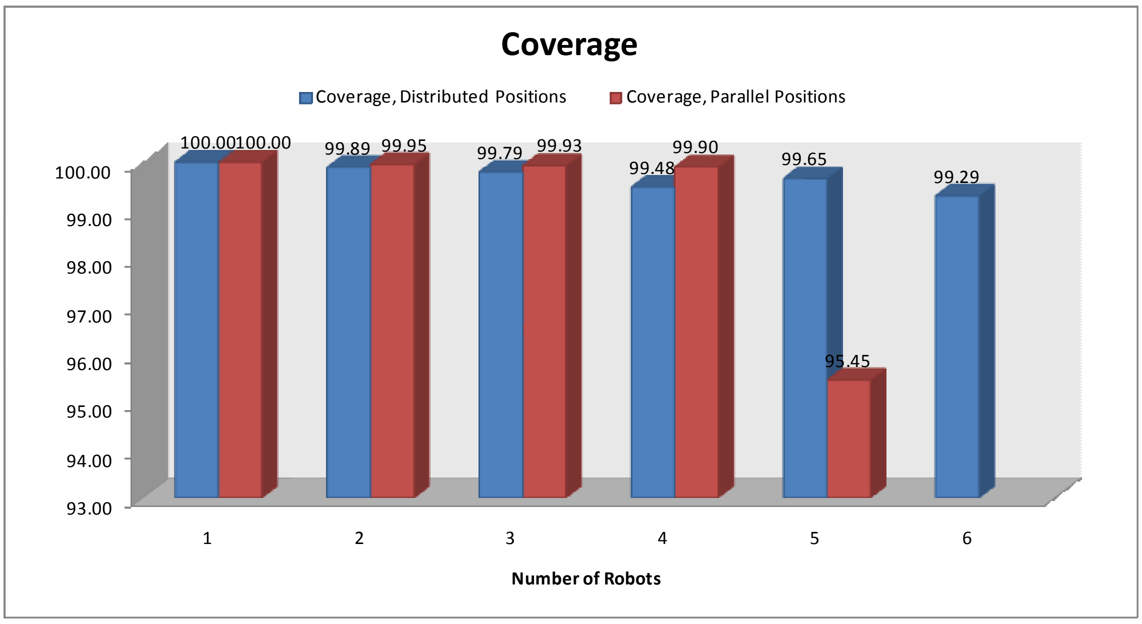


Fig. 6. Total Coverage Percentage results.

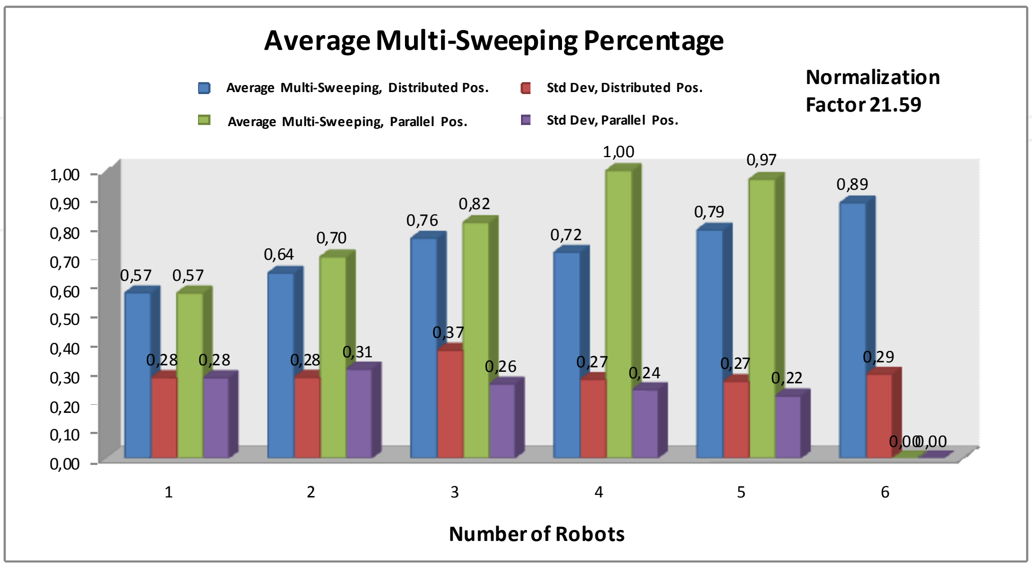


Fig. 7. Average Multi-Sweeping Percentage results.

Coverage: the percentage of covered cells is presented in figure 6. It was observed that in some cases, the robots tend to group in the same zone at the end of the algorithm, causing conflicts and obstructions.

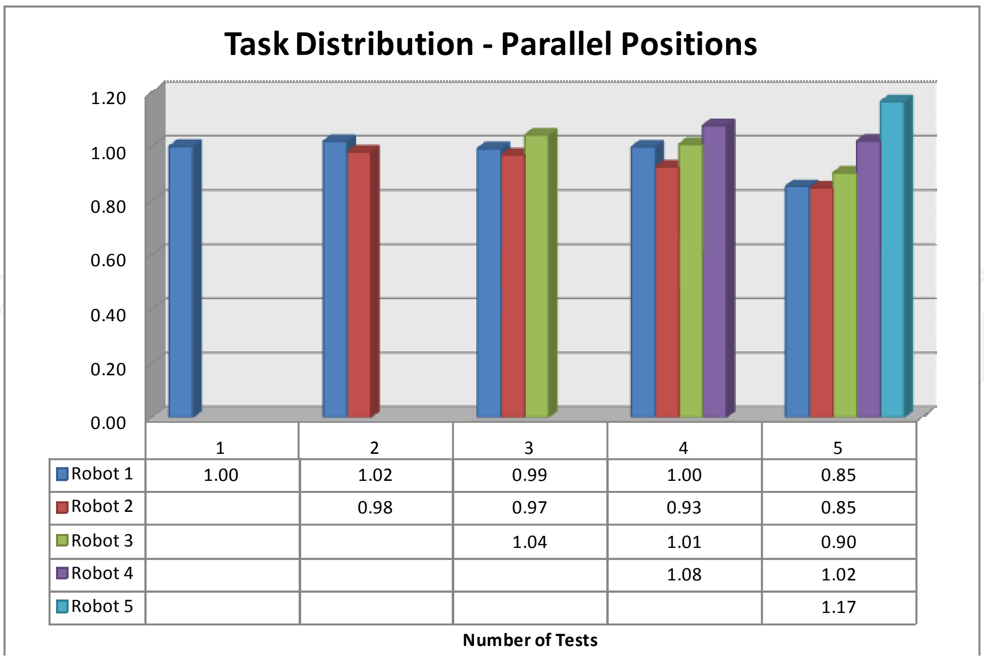
Average of Multi-Sweeping: multi-sweeping shows the number of cells that are covered more than once. Multi-sweeping is expected because the return paths are constructed over visited cells. The average multi-sweeping percentage (figure 7) is higher in the parallel initial position case as robots finish their first spiral sooner, causing the execution of more return paths.

Conflict Management: As a cooperation indicator, the conflict resolution rate indicates the number of conflicts treated in the execution of the algorithm. In most of cases, this indicator is near to 100%.

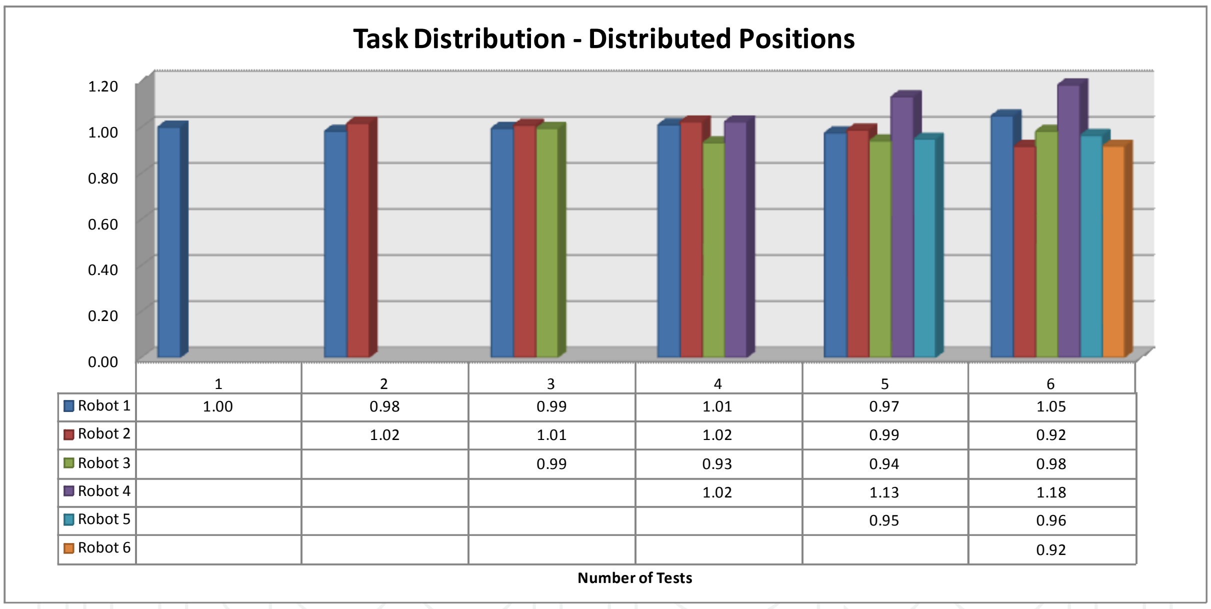
Task Distribution: it is represented by the ratio of covered cells for each robot and the total available cells divided by the number of robots. The average of indicators is near to 1 in all cases, which means that the algorithm is efficient in the task distribution (figures 8a and 8b).

6. Final discussion

In order to cover the entire accessible surface, a sweeping algorithm must eliminate systematically all the known frontier by driving the robot to explore and eventually cover the non-visited regions adjacent to this frontier. Structured trajectories tend to provide a more efficient covering procedure, in terms of reducing the multi-sweeping rate; they also guarantee the detection of the uncovered points nearby the known frontier.



	1	2	3	4	5
🟦 Robot 1	1.00	1.02	0.99	1.00	0.85
🟥 Robot 2		0.98	0.97	0.93	0.85
🟩 Robot 3			1.04	1.01	0.90
🟪 Robot 4				1.08	1.02
🟦 Robot 5					1.17



	1	2	3	4	5	6
🟦 Robot 1	1.00	0.98	0.99	1.01	0.97	1.05
🟥 Robot 2		1.02	1.01	1.02	0.99	0.92
🟩 Robot 3			0.99	0.93	0.94	0.98
🟪 Robot 4				1.02	1.13	1.18
🟦 Robot 5					0.95	0.96
🟧 Robot 6						0.92

Fig. 8. Average Task Distribution in (a)Parallel and (b) Distributed initial positions.

BSA-CM achieves the deployment of the team all over the exploration area by means of backtracking point assignation using the described negotiation mechanism. In fact, robots will be ready to reserve the nearest missions. In this first version of BSA-CM, only homogenous agents have been used; however, it will be easy to extend the algorithm to deal with a team of heterogeneous robots. Coordination and collaboration have been implemented by the proposed distributed mechanism for task assignation. The existence of a task for conflict supervision allows detecting conflicts at early stages, and the mechanism of "role exchange" solves these deadlock situations.

From the point of view of the proposed agency model, the agents are modeled as autonomous entities, which are (a) proactive since they work continuously to achieve their goals following a spiral path or executing a return path, (b) cooperative since the agent's goals depend on team's goals, and (c) each member of the team have a role which defines agent’s responsibilities and partial temporal goals. The system develops relations with the environment and other agents since each robot interact with the world by means of its sensors and elaborate communicative acts with clear semantics in the application context to inform new discoveries in the coverage task. Cooperation under this agency model is evidenced since the purposes of the semantic interactions are task allocation, synchronization between agents and conflict identification and resolution. The protocols designed for the interaction allow the agents to have well-formed conversations including enough information and optimization of the communication channel. In fact, messages in BSA-CM include negotiation of BPs, cell entry protocols and role swapping.

From the point of view of the proposed cooperation model, there are three main aspects to be analyzed: task allocation understood as collaboration, synchronization understood as coordination, and conflict resolution. Collaboration is present as BSA-CM implements a decomposition of the coverage task in simple regions covered by each agent using spiral paths starting from assigned BPs; the negotiation mechanism allows to select the best robot for each particular available BP. Evidence of synchronization is present in the shared replicated map which is constructed incrementally and synchronized as the robots perform their particular spirals. The messages include relevant information via multicast about obstacles and new BPs. Finally, conflict resolution, where the physical space is the conflictive resource, is present in the cell reservation mechanism, which only allows one robot in a cell at any moment; this solution also gives to the agents the possibility of planning future movements taking into account all the team members’ actions. Besides, when a deadlock in return paths is presented as corridors, the system implements a role exchange protocol which solves the space conflict.

The use of an optimistic cost function in the BP negotiation produces relations of competence between the robots. The use of a greedy decision approach generates a greater benefit for the agent in particular and also for the team in general. Robustness, understood as the fulfillment of the task even if there are communications or agent failures, is also assured; in these conditions, the coverage will be performed by all the active robots all over the area; experiments have demonstrated this behavior when there is no communications at all.

Future work includes a complete evaluation of BSA-CM using real robot platforms. Then, the algorithm must be extended to cover also partially occupied cells in a similar way as the basic BSA extension does. Moreover, as the decision of classifying a cell as occupied or free can generate differences between the readings, it is required a method to assign a value of certainty to a particular cell - by means of Bayesian updating model (Martin & Moravec, 1996) for example. It is also important to compare the requirements and indicators with other multi-robot coverage algorithms.

7. References

Menzel, P.(2000). *Robo-sapiens evolution of a new species*. The MIT press, 2000

Ferber, J. (1999). *Multiagent Systems: An Introduction to Distributed AI*, Addison – Wesley Longman. 1ra. ed.

Maes, P. (1994). *Modeling Adaptative Autonomous Agents*, Artificial Life Journal, v. 1, MIT Press.

Rusell, N. (1996). *Artificial Intelligence : A modern approach*, Prentice Hall.

Tolosa G. & Bordignon F. (1999). *Review: Technology of agents of software*, Ci. Inf., Brasilia, v. 28, set./dez.

Wooldridge, M. & Jennings,N.(1998). *Agent Technology – Foundations, Applications, and Markets*. Springer - UNICOM.

Wolldrige, M. (2009) *An introduction to Multiagent Systems*. Ed John Whiley & Sons. 2 ed. ISBN : 978-0470519462

Usategui, J.; Romero, S. & Angulo, I. (1999). *Microbótica: Tecnología, Aplicaciones y Montaje Práctico*. Ed. Thompson. 2 ed. Madrid.

Weiss, G. (2001). *Multiagent Systems: a modern approach to distributed artificial intelligence*. MIT press, ISBN: 0262731312.

Lui, J. & Jianbing, W. (2001). *Multiagent Robotic Systems*. Ed CRC PRESEE. 2001. ISBN : 0-8493-2288-X

Balch, T. & Parker, E. (2002). *Robot Teams*. AK Peters, Canada. ISBN : 1-56881-115-1.

Miriad E. (1992). Approcher la notion de collectif. *En Actes des journées multiagents du PRC-IA*, Nancy.

Jung, D. (1998). *An Architecture for Cooperation among Autonomous Agents*. Intelligent Robotics Laboratory, Department of Computer Science, University of Wollongong.

Murphy, R. (2000). *Introduction to AI Robotics*. Ed The MIT Press. ISB : 0-262-13383-0

Choset, H. (2004). *Principles of robot motion theory, algorithms, and implementation*. MIT Press. ISBN 0-262-03327-5

Hopgood, A. (2001). *Intelligent Systems for Engineers and Scientist*. Boca Raton, Florida ; London : 2nd ed CRC.

Tilde, M.W. (1996) Yuma proving grounds Automatic Uxo Detection using Biomorphic Robots. *Test Teschnology Symposium*. US Army Test & Evaluation Com.

Doty K. & Harrison R. (1993). Sweep Strategies for a Sensory-Driven, Behavior-Based Vacuum Cleaning Agent. *AAAI Fall Symposium Series.*

Choset, H. & Pignon, P. *Coverage Path Planning: The Boustrophedon Cellular Decomposition*. unpublished

Zelinsky A. et al. (1992). A mobile robot exploration algorithm. *Robotics and Automation, IEEE Transactions on*. Volume 8, Issue 6, Page(s):707 - 717

Pirzadeh, A. & Snyder W. (1990). A unified solution to coverage and search in explored and unexplored terrains using indirect control. *Robotics and Automation, Proceedings. IEEE International Conference on*. Page(s):2113 - 2119 vol.3.

González E. et al. (1996). Complementary Regions: a Surface Filling Algorithm. *ICRA96*, pp. 909-914..

Gabriely Y.; & Rimon E. (2002). Spiral-STC: an on-line coverage algorithm of grid environments by a mobile robot. *IEEE International Conference on Robotics and Automation,*. Proceedings. ICRA 2002. Volume 1, Page(s):954 – 960.

Gonzalez, E., et al. (2005). BSA: A Complete Coverage Algorithm. *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. Pages: 2040 – 2044

Martin, M.C. & Moravec, H. (1996) *Robot Evidence Grids*. The Robotics Institute. Carnegie Mellon Univ.. USA, CMU-RI-TR-96-06.

Choset, H., et al. (2004) Towards sensor based coverage with robot teams. *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*. Volume 4, Page(s):3462 - 3468 Vol.4

Simmons R. Et al. (2000). Coordination for multirobot exploration and mapping. *Proceedings of the National Conference on Artificial Intelligence. AAAI*, 2000

Dias, M.B. et al. (2006). Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, pages: 1257-1270..

Wagner, I.A.; Lindenbaum, M. & Bruckstein, A.M. (1999). Distributed covering by ant-robots using evaporating traces. *Robotics and Automation, IEEE Transactions on*, Volume 15, Issue 5, Page(s):918 - 933.

Butler Z.J.; Rizzi, A.A. & Hollis, R.L. (2000). Cooperative Coverage of Rectilinear Enviroments. *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, Volume 3, 24-28 Page(s):2722 - 2727 vol.3

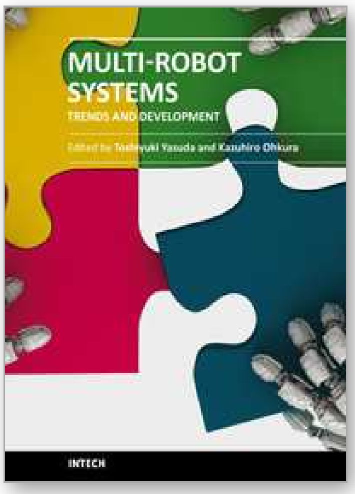
Howard; Mataric & Sukhtme. (2002). An Incremental Deployment Algorithm for Mobile Robot Team. *IEEE/RSJ International conference in Robotics and Intelligent Systems.* Switzerland.

Gerlein E., Gonzalez E.. (2009) BSA-CM: A Multi-Robot Coverage Algorithm. *2009 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*.

Gonzalez E., et.al. (2008) *Development of Applications based on Multi-Agent Systems*. Editorial Javeriana, Colombia 2008.

Rizzi, A.; Gowdy, J. & Hollis, L.. (1997) Contact sensor-based coverage of rectilinear environments. *Proc. Of IEEE Int'l. Conf. on Robotics and Automation*. Pp 1511-1516.

Zlot R., et al. (2002);. Multi-robot exploration controlled by a market economy. Conference on Robotics and Automation, 2002. Proc. ICRA '02. IEEE International.


MULTI-ROBOT SYSTEMS
TRENDS AND DEVELOPMENT
Edited by Toshiyuki Yasuda and Kazuhiro Ohkura
INTECH