Multi-robot coordination strategies for exploration

Ket-ziquel Hernández, Abraham Sánchez, Maria A. Osorio†, Alfredo Toriz P.‡
and Francisco Sosa

Computer Science Department
†Chemical Engineering Department
Benemérita Universidad Autónoma de Puebla
‡Robotics Department
LIRMM, UMR 5506 - CC477
`ket2107@hotmail.com, asanchez@cs.buap.mx, alfredo.torizpalacios@lirmm.fr`

(Paper received on November 28, 2010, accepted on January 28, 2011)

Abstract. The approach to explore an unknown environment by multiple robots is a parallelization of the SRT method, which is based on the random generation of robot configurations within the local safe area detected by the sensors. In this paper, we propose several coordination strategies to solve the cooperative exploration problem. We focus our attention in the cooperative policy strategy, which is completely decentralized as each robot decides its own motion by applying some rules only on the locally available information. Simulation results in various environments are presented to compare the performance of proposed coordination strategies.

1 Introduction

Multiple robots are increasingly used in different applications to cooperatively solve complex tasks. Many successful robotic systems use maps of the environment to perform their tasks. There are numerous studies to find efficient ways for exploring and creating maps for an unknown environment [14], [7], [15], [10], [8], [4]. Unfortunately, most of the studies deals with single robots. However, exploring an unknown environment with single robots has several disadvantages, e.g., real-time applications with single robots takes more time than using multi-robots. Besides, single robots can not produce accurate maps like multi-robots. The only advantage of using a single robot is the minimization of the repeated coverage. Even though repeated coverage among the robots decreases the mission's efficiency, some amount of repeated coverage is a desirable situation for better efficiency, this better efficiency can be achieved by coordination among robots. If multi-robots can explore an unknown area faster than a single robot, there is a very important question: How can we coordinate the behavior of robots in the unknown area?.

This paper presents a method to explore an unknown environment by multi-agent robots, the Multi-SRT approach. This method is a parallelization of the SRT (Sensor-based Random Tree) idea, which was presented in [8]. The present

(C) C. Zepeda, R. Marcial, A. Sánchez
J. L. Zechinelli and M. Osorio (Eds)
Advances in Computer Science and Applications
Research in Computing Science 53, 2011, pp. 89-99

paper builds on the previous work presented in [11], [12], [6]. The basics of the Multi-SRT method are briefly presented in Section II. The proposed multi-agent robots coordination are detailed in Section III. Simulation results in different environments are discussed in Section IV. Finally, conclusion and future work are detailed in Section V.

2 The multi-SRT method

Consider a population of n identical robots. Each robot is equipped with a ring of range finder sensor or a laser range finder, the sensory system provides the local safe region $S(q)$. The robots move in a planar workspace, i.e., $\mathbb{R}^2$ or a connected subset of it; the assumption of planar workspace is not restrictive. Each robot is a polygon[1] or another shape subject to non-holomic constraints. Each robot also knows its configuration q, one can eliminate this assumption by incorporating a localization module in the method. The robots know its ID number and each robot can broadcast within a communication range R_c the information stored in its memory (or relevant portions of it) at any time. The robot ID number is included in the heading of any transmission. The robot is always open for receiving communication from other robots inside R_c.

The exploration algorithm for each robot is shown in Figure 1. First, the procedure BUILD_SRT is executed, i.e., each robot builds its own SRT, $\mathcal{T}$ is rooted at its starting configuration q_{init}. This procedure terminates when the robot can not further expand $\mathcal{T}$. Later, the robot executes the SUPPORT_OTHERS procedure, this action contributes to the expansion of the SRTs that have been built by others robots. When this procedure finishes, the robot returns to the root of its own tree and finishes its exploration. For more details of the approach, one can consult the work [12].

```
**BUILD Multi-SRT**($q_{init}$)
1 $\mathcal{T}.init(q_{init})$
2 BUILD_SRT($q_{init}.\mathcal{T}$);
3 SUPPORT_OTHERS($q_{init}$);
```

Fig. 1. The Multi-SRT algorithm.

In each iteration of the BUILD_SRT, the robot uses all available information partially collected by itself and partially gained through the communication with other robots. The procedure SUPPORT_OTHERS can be divided into two major phases, which are repeated over and over again. In the first phase, the robot picks another robot to support it in his exploration, or, more precisely, another tree

---

[1] Polygonal models make it possible to efficiently compute geometric properties, such as areas and visibility regions.

that helps it to expand (there may be more than one robot acting on a single tree). In the second phase, the selected tree is reached and the robot tries to expand it, tying subtrees constructed by the procedure BUILD_SRT. The main cycle is repeated until the robot has received confirmation that all the other robots have completed their exploration [11]. Figure 2 shows two different views of the execution of the Multi-SRT algorithm in an environment that contains 10 nonholonomic mobile robots.

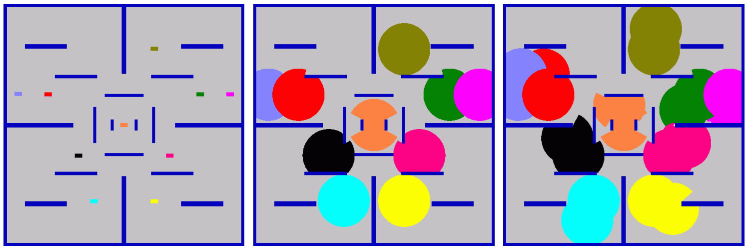

Fig. 2. Snapshots showing the execution of the Multi-SRT algorithm.

```
BUILD_SRT($q_{init}, \mathcal{T}$)
$q_{act} = q_{init}$;
do
  BUILD_AND_WAIT_GPR();
  $S(q_{act}) \leftarrow$ PERCEIVE($q_{act}$);
  ADD($\mathcal{T}, (q_{act}, S(q_{act}))$);
  $\mathcal{G} \leftarrow$BUILD_GER();
  $\mathcal{F}(q_{act}) \leftarrow$ LOCAL_FRONTIER($q_{act}, S(q_{act}), \mathcal{T}, \bigcup \mathcal{T}_i$);
  $q_{target} \leftarrow$ PLANNER($q_{act}, \mathcal{F}(q_{act}), q_{init}$);
  if $q_{target} \neq NULL$
    if $|\mathcal{G}| > 1$
      $(\mathcal{G}_f, \mathcal{G}_u) \leftarrow$ CHECK_FEASIBILITY($\mathcal{G}$);
      if $\mathcal{G}_u \neq \emptyset$
        $q_{target} \leftarrow$ COORDINATE($\mathcal{G}_f, \mathcal{G}_u$);
    $q_{act} \leftarrow$ MOVE_TO($q_{target}$);
while $q_{target} \neq NULL$
```

Fig. 3. The BUILD_SRT procedure.

The local coordination procedures implemented in the proposal guarantees that the collective motion of the robots are feasible from the collision viewpoint. The approach does not need a central supervision. The selection of exploration actions by each robot is spontaneous and it is possible on the basis of the available information.

3 Coordination strategies

Since exploration task requires cooperation and coordination among robots, the achievement of the task will be accidental if robots work independently. This task can not be done unless robots cooperate and coordinate their behaviors. Therefore, it is necessary to have cooperation strategies that allow multiple robots to help each other in the problem solving process.

To solve a multi-robot task, either centralized or decentralized (distributed) approaches can be used. A centralized model uses a powerful robot to plan and schedule the subtasks for every robot. This control robot has a global knowledge concerning the environment and the problems. It can deliberately plan for better performance. On the other hand, a distribute approach decreases design complexity and cost, while increasing the reliability. Robots are autonomous and equal. A robot plans for itself and communicates with the others in order to accomplish the global task. Since every robot interacts directly with the environment, it is reactive. However, each robot has only local knowledge of the task and the environment. Hence, it cannot make the best decision of the global task alone. Furthermore, negotiation and social cooperation rules for conflict resolution are required to coordinate among them.

In this work, several strategies were utilized to solve multi-robot exploration tasks, in the next paragraphs we introduce two new strategies, a blackboard approach and a decentralized cooperative policy for conflict resolution. The two originally proposed strategies in the work presented in [12] were also considered (i.e., coordination via arbitration and coordination through replanning [11]).

A blackboard system in general, is a distributed, opportunistic approach to system design. It is characterized by a set of knowledge sources that can communicate with each other via an area of global memory called a blackboard. Each knowledge source is designed to solve a specific component of the problem that the system is presented with. The blackboard is a global section of memory that is accessible to all of the knowledge sources. The blackboard contains the data, as well as partial solutions to the problem at hand. In a robotic system, the blackboard could be seen as a representation of the world state, through sensor input, actuator positions, world maps, and other pertinent information. The set of knowledge sources comprises the problem-solving component of the system. Each of the knowledge sources is tailored to a specific function.

Frazzoli et al. [5], proposed a novel policy for steering multiple vehicles, the policy rests on the assumption that all agents are cooperating by implementing the same rules. They mentioned that their policy is completely decentralized, as each robot decides its own motion by applying those rules only on the locally available information processed by each robot. Their policy applies to systems in which new mobile robots may enter the scene and start interacting with existing ones at any time, while others may leave. The proposed spatially decentralized control policy is based on a number of discrete modes of operation [9].

In order to explain the rules on which this coordination strategy is based, it is necessary to define the suitable annotation to refer us to the robot position at certain time. A configuration describes the pose of the robot, i.e., it can be represented using two parameters (x, y). We call plan to a pair of positions (g_s, g_g), which are a start configuration and a goal configuration, respectively. One can define a plan as follows: plan(g_s, g_g) is a safe movement of a robot in an environment, where g_s is the current position and g_g is the candidate position. During the exploration time, each robot collects all starting points in a set G_s and their goal positions in a set G_g. Let $G_s = \{g_s, s = 1, ..., n\}$ the set of all start robots configurations at certain time and $G_g = \{g_g, g = 1, ..., n\}$ the set of all goal robots configurations at certain time, where i denotes the ID robot. We can mention the following rules of the proposed policy, which can guarantee to the system to be collision-free.

Admissibility. One can consider an environment in which new robots may issue a request to enter the scenario at an arbitrary time and with an arbitrary plan, consisting of a start and goal configurations. It is important to have conditions to efficiently decide on the acceptability of a new request. The new proposed plan is compatible with the properties $\mathbf{P_1}$ and $\mathbf{P_2}$.

> $\mathbf{P_1}$: A configuration set $G_s = \{g_s, s = 1, ..., n\}$ is unsafe for the policy ζ, if there exist a set of target $G_g = \{g_g, g = 1, ..., n\}$ such that ζ leads to a collision.
> $\mathbf{P_2}$: A target configuration set $G_g = \{g_g, g = 1, ..., n\}$ is blocking for the policy ζ, if there exist a set of configurations $G_s = \{g_s, s = 1, ..., n\}$ from which ζ leads to a dead-lock or live-lock.

A plan (G_s, G_g) is admissible if it verifies the predicate $\neg\mathbf{P_1}(G_s) \wedge \neg\mathbf{P_2}(G_g)$, i.e., there are no collisions with the robots plans in the environment.

Well-posedness. The first step of this coordination policy is to verify that each robot is a well-posed dynamical system, i.e., a solution exists and is unique, for all initial conditions within a given set. In other words, from the beginning and during the exploration process of the environment, each robot does not have to invade the safe regions of others robots. Next figure illustrates this rule. The safe region is a geometrical form created from its sensors.

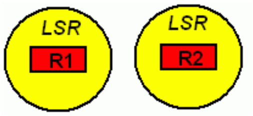

Fig. 4. The safe regions of each robot initially do not overlap between them.

This rule is defined by the next theorem, for more details, see [9].

Theorem 1 *The system is well-posed, for all initial conditions in which the interiors of local safe regions are disjoints.*

Safety. This rule of the cooperative policy proposes that the safe regions of robots are not overlapped during the exploration process; if at some given time there exists an overlapping, the robot does not have to advance, i.e., the robot searches for a new candidate position, in case of not finding it, the robot will remain in its position. The figure shows overlaps of safe regions of two robots, this problem is corrected with the safety rule.

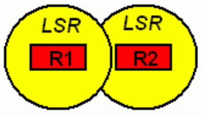

Fig. 5. The safe regions of near robots never must be overlapped.

According to the previous description, this safety rule is related with the property $\mathbf{P_1}$, from which the following theorem is derived [9].

Theorem 2 *If the safe regions (SR) of at least two robots overlap, property* $\mathbf{P_1}$ *is verified. In this case, a backtracking policy or a replanning one is implemented. Since robots always are within their local safe region, it is possible to be assured that the system is collision-free.*

Liveness. Liveness is related with the property $\mathbf{P_2}$, it is based in the definition of a condition that maintains separated the safe regions, which are associated to the robots goal configurations. In other words, the liveness of a robot is in charge of maintaining to certain distances the robots goal positions to avoid overlap of these regions and with this possible collisions. This rule is important since it allows to maintain a considerable distance between the robots goal positions, in addition, aid to avoid live-lock, see the following figure.

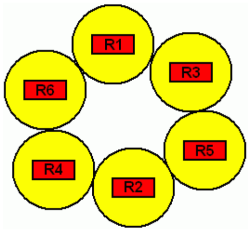

Fig. 6. Live-lock during the exploration process.

Since it is important to avoid live-lock during the exploration process, we can define the following property.

$\mathbf{P_3}$: A configuration set $G_g = \{g_g, g = 1, ..., n\}$ is clustered if the distance between goals configurations is smaller than the radius of the safe region.

From the previous property the following theorem is derived, the proof is detailed in [9].

Theorem 3 *The property* $\mathbf{P_2}$ *is valid for the coordination policy if the property* $\mathbf{P_3}$ *is also valid.*

In other words, the sparsity of goal configurations is a necessary condition to rule out the possibility of blocking executions in this policy, a sufficiency condition is presented in the following theorem.

Theorem 4 *Consider two mobile robots such that the center of the safe regions in final configurations are at distance larger than* $2d + 4$*.*

where d is the Euclidean distance between two configurations.

This policy allows the mobile robots to reach their final destinations in finite time, from all initial conditions such that the safe regions are disjoint. Following the rules previously discussed, we propose the following algorithm for the coordinated exploration with multi-agent robots in unknown environments, see figure 7. It is possible to note that the new algorithm is similar to the original multi-SRT algorithm, i.e., we only adapted it for the cooperative policy as the coordination strategy among robots.

Two procedures have been added to the algorithm: CHECK_POSEDNESS and CHECK_SAFETY_LIVENESS. The first procedure is in charge of checking that the robots safe areas (LSRs) must be disjointed and the robots poses must be to at least a distance of two times the perception radius. The second procedure is in charge of reviewing at each step that the target configurations are not within the local safe regions of other robots, this rule is carried out with a query, in which the goal configuration is searched within the safe local regions of other robots, in the case of that a target configuration exists within the areas, it is necessary to execute again a coordination with the procedure COORDINATE($\mathcal{G}_f, \mathcal{G}_u$), which only realises a search of a new target. The procedure CHECK_SAFETY_LIVENESS also verifies that a blocking does not exist, this is obtained with a condition in the target positions, which must be to less than a distance of two times the perception radius.

4 Simulation results

The tests were performed on a Celeron © 430 processor-based PC running at 1.80 Ghz with 2 GB RAM. The strategies were implemented in Visual C++, taking advantage of the MSL library's structure and its graphical interface[2]. The GPC library developed by Alan Murta was used to simulate the sensors perception systems[3]. The polygonal representation facilitates the use of the GPC

---

[2] http://msl.cs.uiuc.edu/msl/

[3] http://www.cs.man.ac.uk/∼toby/alan/software/

```
COOPERATIVE_POLICY($ID, \mathcal{T}_i, q_{act}, \mathcal{S}(q_{act})$)
$q_{act} = q_{init}$;
CHECK_POSEDNESS($q_{act}$)
loop
  $S \leftarrow$ PERCEPTION($q_{act}$);
  ADD($\mathcal{T}, (q_{act}, S)$);
  $\mathcal{G} \leftarrow$BUILD_GER();
  $\mathcal{F}(q_{act}) \leftarrow$ LOCAL_FRONTIER($q_{act}, S(q_{act}), \mathcal{T}, \bigcup \mathcal{T}_i$);
  $q_{target} \leftarrow$ PLANNER($q_{act}, \mathcal{F}(q_{act}), q_{init}$);
  UPDATE_INFORMATION($ID, \mathcal{T}_i, q_{act}, \mathcal{S}(q_{act})$);
        if $|\mathcal{G}| > 1$
          $(\mathcal{G}_f, \mathcal{G}_u) \leftarrow$ CHECK_SAFETY_LIVENESS($\mathcal{G}$);
          if $\mathcal{G}_u \neq \emptyset$
            $q_{target} \leftarrow$ COORDINATE($\mathcal{G}_f, \mathcal{G}_u$);
        $q_{act} \leftarrow$ MOVE_TO($q_{target}$);
while $q_{act} = q_{init} \wedge \mathcal{F}(q_{act}) = \emptyset$
```

Fig. 7. The cooperative policy algorithm.

library for the perception algorithm's simulation. If S is the zone that the sensor can perceive in absence of obstacles and SR the perceived zone, the SR area is obtained using the difference operation of GPC between S and the polygons that represent the obstacles.

One can consider two possible initial deployments of the robots. In the first, the robots are initially scattered in the environment; and the second, the exploration is started with the robots grouped in a cluster. Since the Multi-SRT approach is randomized, the results were averaged over 10 simulation runs. We consider that an increment of the number of evenly deployed robots corresponds to a decrement of the individual areas they must cover. When the robots are far apart at the start, they can exchange very little information during the exploration process.

Figures 8 and 9 illustrate the Multi-SRT and explored regions with clustered and scattered starts respectively. We can see the difference when the robots are evenly distributed at the start of are clustered. At the end of the exploration process, the environment has been completely explored and the SRTs have been built. In these figures, one can observe that each robot built its own SRT and when one of them finished, this entered the support other phase.

Exploration time for teams of different cardinality are shown in Figures 10 and 11, both in the case of scattered and clustered starts. In theory, when the number of robots increases, the exploration time would quickly have to decrease.

Agent coordination and communication are important issues in designing decentralized agent systems. Various communication strategies are presented in this paper: blackboard, replanning, arbitration and cooperative policy, as

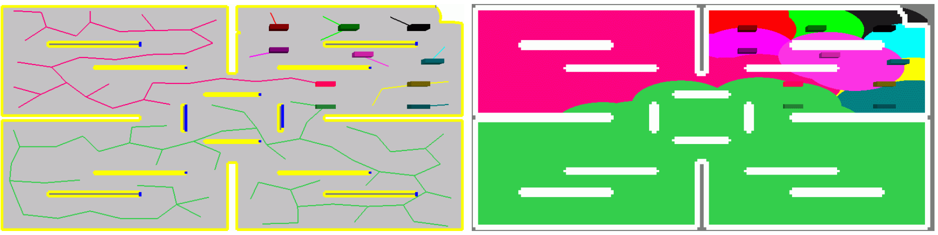

Fig. 8. The Multi-SRT and explored regions with clustered starts with a team of 10 robots.

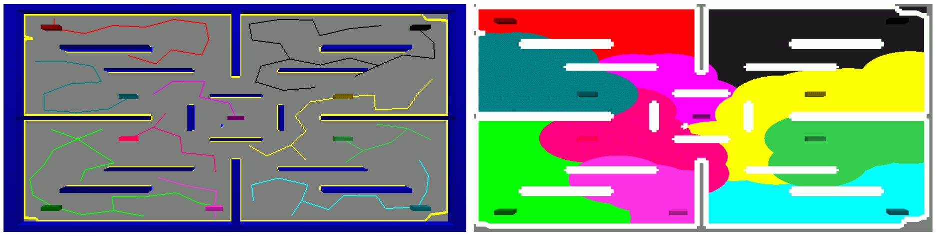

Fig. 9. The Multi-SRT and explored regions with scattered starts with a team of 10 robots.

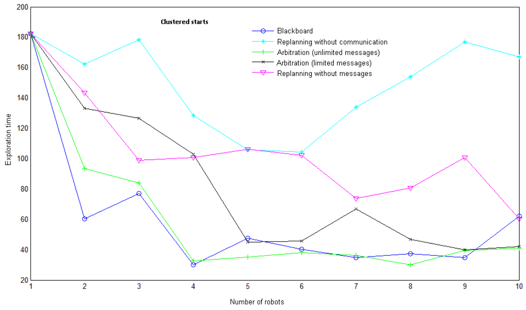

Fig. 10. Exploration time versus number of robots for the clustered starts situation.

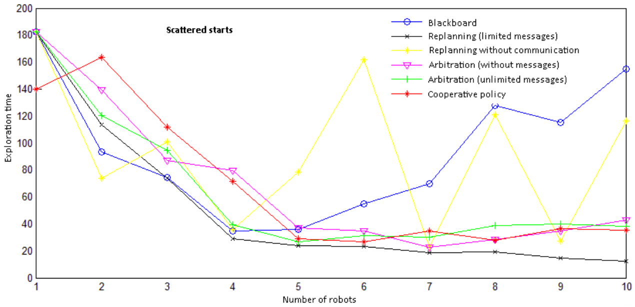

Fig. 11. Exploration time versus number of robots for the scattered starts situation.

part of the coordination for environments exploration, which, by nature, are implemented in limited and unlimited communication ranges.

The graphic generated with the exploration times obtained, show that for those environments with a scattered initial configuration, the communication strategies more efficient are the limited communication with messages and the limited communication with no messages. The disadvantage of the limited communication without messages strategy, is that it will require more re-exploration in the presence of more robots, i.e., increasing the exploration time, unlike the limited communication with messages that prevents re-exploration. In the graphic of the initial cluster configuration, it can be seen that, as more sophisticated and less centralized is the communication between the robots, as more optimal are the results corresponding to the exploration and the resolution of conflicts. The analysis of the results obtained for the case of the initial configuration in cluster shows that any communication strategy either limited or unlimited is useful for resolving conflicts related to collisions and to carry out the exploration efficiently.

The communication strategy with blackboard tends to be sluggish when the number of robots in the environment is increased, because the structure used as a board is shared and only one robot at a time can access it; if more than a robot wants to access the board at the same time, it must wait in line for a turn. The cooperative policy shows the best results, even in those environments considered difficult to explore. From a particular point of view the good performance of this policy is because, it initially performs a quick exploration of the environment. This can be verified after analyzing the implementation of our approach with all the coordination strategies, and compare their performance in the graphic obtained with the strategy of cooperative policy, in the support others phase that the robots use to complement the exploration.

Finally, it can be concluded that the integration of a communication strategy in the robots, as a way of coordination to avoid conflicts, is very useful in the task of environment exploration, because it can help them to share information in order to avoid conflicts related to collisions and, in some cases, to prevent re-exploration areas in the most important phase, when a good communication strategy can contribute in reducing the exploration time.

5 Conclusions and future work

Exploration using multiple robots is characterized by techniques that avoid tightly coordinated behavior. The implemented policy gives rise to a hybrid system, which can be shown to be well posed and safe, if the initial configurations satisfy a rather nonrestrictive condition. Through the examples, we can affirm that the policy is spatially decentralized and its complexity is bounded regardless of the number of agents.

Exploration and localization are two of capabilities necessary for mobile robots to navigate robustly in unknown environments. A robot needs to explore in order to learn the structure of the world, and a robot needs to know its own location in order to make use of its acquired spatial information. However, a problem arises with the integration of exploration and localization simultaneously. The integration of a localization module into the exploration process based on SLAM techniques will be an interesting topic for a future research. We can also consider an extension of the Multi-SRT exploration method, where the robots constantly maintain a distributed network structure.

References

1. W. Burgard, M. Moors and F. Schneider, "Collaborative exploration of unknown environments with teams of mobile robots", *Plan-Based Control of Robotic Agents, LNCS*, Vol. 2466, (2002)
2. Y. Cao, A. Fukunaga and A. Kahng, "Cooperative mobile robotics: Antecedents and directions", *Autonomous Robots*, Vol. 4, (1997)1-23
3. G. Dudek, M. Jenkin, E. Milios and D. Wilkes, "A taxonomy for multi-agent robotics", *Autonomous Robots*, Vol. 3, (1996) 375-397
4. J. Espinoza L., A. Sánchez L. and M. Osorio L., "Exploring unknown environments with mobile robots using SRT-Radial", *IEEE Int. Conf. on Intelligent Robots and Systems*, (2007) 2089-2094
5. E. Frazzoli, L. Pallotino, V. G. Scordio and A. Bicchi, "Decentralized cooperative conflict resolution for multiple nonholonomic vehicles", *Proc. of the American Institute of Aeronautics and Astronautics Conf.*, (2005)
6. K. Hernández Guadarrama, "Estrategias de coordinación para la exploración con multi-agentes robóticos", *Master Thesis*, FCC-BUAP (in spanish), (2010)
7. J. Ko, B. Stewart, D. Fox, K. Konolige and B. Limketkai, "A practical, decision-theoretic approach to multi-robot mapping and exploration", *IEEE Int. Conf. on Intelligent Robots and systems*, (2003) 3232-3238
8. G. Oriolo, M. Vendittelli, L. Freda and G. Troso, "The SRT method: Randomized strategies for exploration", *IEEE Int. Conf. on Robotics and Automation*, (2004) 4688-4694
9. L. Pallottino, V. G. Scordio, A. Bicchi and E. Frazzoli, "Decentralized cooperative policy for conflict resolution in multi-vehicle systems", *IEEE Transactions on Robotics*, Vol. 23, No. 6, (2007) 1170-1183
10. R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun and H. Younes, "Coordination for multi-robot exploration and mapping", *17th Conf. of the American Association for Artificial Intelligence*, (2000) 852-858
11. A. Toriz Palacios, "Estrategias probabilisticas para la exploración cooperativa de robots móviles", *Master Thesis*, FCC-BUAP (in spanish), (2007)
12. A. Toriz P., A. Sánchez L. and M. A. Osorio, "Coordinated multi-robot exploration with SRT-Radial", *LNAI 5290, Springer-Verlag*, (2008) 402-411
13. A. Toriz P., A. Sánchez L. René Zapata and M. A. Osorio, "Building feature-based maps with b-splines for integrated exploration", *LNAI 6433, Springer-Verlag*, (2010) 562-571
14. B. Yamauchi, " Decentralized coordination for multirobot exploration", *Robotics and Autonomous Systems*, Vol. 29, (1999) 111-118
15. R. Zlot, A. Stenz, M. Dias and S. Thayer, "Multi-robot exploration controlled by a market economy", *IEEE Int. Conf. on Robotics and Automation*, (2002) 3016-3023