An Integrated Task Allocation Approach for Multi-Robot Navigation in Realistic Scenarios

Antidio Viguria and Ayanna M. Howard
Human-Automation Systems Lab
School of Electrical and Computer Engineering
Georgia Institute of Technology
30332 Atlanta, USA
antidio@gatech.edu and
ayanna.howard@ece.gatech.edu

***Abstract*— In this paper, we discuss a distributed algorithm for task allocation that solves the Initial Formation Problem within the multi-robot domain. The algorithm has been integrated in a multi-robot architecture that couples the task allocation behavior with a path planning and navigation module. Analysis of the efficiency of the tak allocation methodology is provided that compares results from a realistic scenario that simulates achieving multiple science measurements within an Arctic terrain environment.**

I. INTRODUCTION

In future science exploration missions, there is a desire to send multiple, instrumented rovers to scientific sites of interest to expand our understanding of both the history and future of life. Mars exploration missions are focused on finding signs of life to expand our comprehension of where life began. Earth exploration missions are focused on resolving theories on how life evolved and how it might be effected in the future. These mission examples all have one common theme – human scientists and autonomous rovers must work together to navigate in extreme environments in order to collect scientific measurements of interest. This type of problem, that of navigating to multiple science sites using multiple rovers, can be cast as a task allocation problem.

In the last years, different approaches have been used to solve the general task allocation problem: centralized [2], hybrid [3] and distributed [18]. However, there are other types of problems that cannot be solved with these algorithms, for example, the Initial Formation Problem [8]. This type of problem becomes really important within the field of formation control [9] where using local information and control laws, the distributed algorithm is able to drive a given formation error to zero. However, as it is stated in [10], these algorithms require a first step that assigns the robots to the formation positions while taking into account their initial positions, i.e., answer the question who goes where? Usually this problem has been solved using centralized solutions such as the Hungarian method [11], since the Initial Formation Problem can be viewed as a classical job assignment problem where robots are the workers and tasks are the jobs to be executed by those workers. However, this type of solution requires that all the robots have to communicate between each other and has all the disadvantages related to centralized systems: low fault tolerant, computational complexity and slow response for dynamic changes in the environment. Furthermore, it is not possible to take advantage of all the good characteristics related to distributed algorithms if part of your problem has to be solved in a centralized way.

Regardless of the fact that the multi-robot task allocation problem has been studied for the last decade, most of the algorithms have been tested assuming idealistic simulations and only considering waypoint tasks where the cost is just the euclidean distance. Only a few have been tested on real robots [1], [4], [7] and they usually deal with proving of concepts.

For these reasons, it is important to come up with an algorithm that solves the Initial Formation Problem while taking into account the realisms of operating in extreme environments. As such, in this paper, we discuss our approach to the Initial Formation Problem using a distributed based approach [5] and analyze its efficiency in a realistic scenario. The paper is organized as follows. In the next section, a basic market-based (BS) algorithm that solves the Initial Formation Problem will be explained. Also, different modifications of the BS algorithm that improve its results will be addressed. Section III discusses the integration of task allocation, path planning and navigation. The complete robotic system, the relations between the different parts and the algorithms used are also explained in Section III. In Section IV, different simulations are shown and discussed. Firstly, the task allocation algorithms are simulated with a simple simulator that only considers the euclidean distance as the cost and in a world without obstacles. Secondly, these results are compared with the ones obtained in an obstacle-strewn world in which the task allocation algorithms are integrated with different path planning algorithms (A^* and RRTs). Finally, conclusions and future work are discussed in Section V.

II. TASK ALLOCATION ALGORITHM

A market-based approach has been chosen to solve the Initial Formation Problem using the Contract Net Protocol [17] where two roles are played dynamically by robots: auctioneer and bidders. The auctioneer is the agent in charge of announcing the tasks and selecting the best bid from all

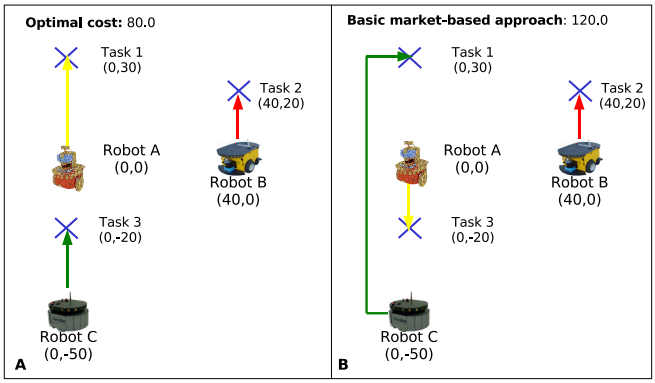

Fig. 1. Difference in cost between the optimal allocation and the one obtained with the basic market-based algorithm.

the received bids. In our case the best bid is the one with the lowest cost.

A. BS: Basic Market-Based Approach

We have implemented a basic algorithm [8] where the cost for bidding is equal to the distance from the robot to the task. The basic idea is that each robot can only own one task, so it will keep the task with the lowest cost. If it wins a new task that has a lower cost than the one already won, it will sell the old task to the robot with the best bid but worse than its own bid. The best bid worse than the robot's bid is selected in order to avoid infinite loops in the negotiation. This scenario could happen when two robots have the best bids for at least three tasks [8].

There are situations where this algorithm does not obtain good results which usually happens when a robot wins a task that is the worst one for its own interest, as can be seen in Figure 1. In this example, the global cost obtained with the market-based algorithm is 66.67% greater than the optimal allocation.

B. RTMA: Robot and Task Mean Allocation algorithm

In order to solve the initial formation problem, the task allocation algorithm has to solve two main problems:

- How do I calculate the bid for a certain task?
- If I won more than one task, how do I determine which one to keep?

The RTMA algorithm tries to improve the original algorithm in these two aspects, while keeping the advantages of the market-based approach: fault tolerance, independent from the number of robots and high adaptation to changes in the environment using reallocations. Firstly, it chooses in a more clever way the task that must be kept when a robot wins more than one task. This is accomplished using additional knowledge available to the system. Instead of keeping the task with the smallest cost to the robot, the task with highest difference between the cost to the robot and the mean of its costs to all the robots is selected. In other words, suppose that there are a finite number of tasks T and robots N and robot R_k has won tasks T_i and T_j. In this case, robot R_k will keep task T_i if and only if:

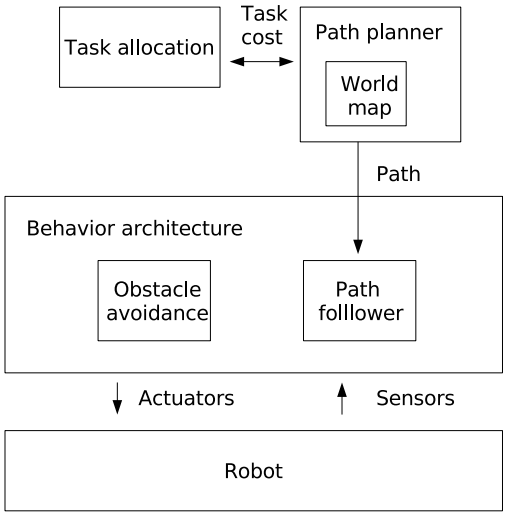

Fig. 2. Scheme that shows the integration of a task allocation algorithm in a complete system ready to be used in a real world application. The path planning algorithm is used to calculate the cost of the tasks and as an input for the path follower algorithm which is combined with obstacle avoidance using the DAMN architecture.

$$\sum_{l=1}^{N} \frac{C(R_l, T_i)}{N} - C(R_k, T_i) > \sum_{l=1}^{N} \frac{C(R_l, T_j)}{N} - C(R_k, T_j) \quad (1)$$

where $C(R_a, T_b)$ is the cost to execute task T_b by robot R_a.

Secondly, the cost function is changed. In the original algorithm the cost function used to calculate the bid for a certain task is the distance between the robot and the task. However, in this improved algorithm the cost function is the difference between the distance of the robot and the task minus the mean of the distances between that robot and all the tasks, i.e.:

$$C(R_i, T_j) = D(R_i, T_j) - \sum_{k=1}^{N} \frac{D(R_i, T_k)}{N} \quad (2)$$

where $C(R_i, T_j)$ is the cost function for robot R_i and task T_j and $D(R_a, T_b)$ is the distance between robot R_a and task T_b.

When one robot wins two tasks, instead of comparing the distances to choose the closest one, it will compare the costs using the new cost function and it will select the task with the lowest cost for itself.

III. INTEGRATION OF THE TASK ALLOCATION ALGORITHM WITHIN A ROBOTIC SYSTEM

We have implemented our task allocation algorithms using a multi-robot architecture [13] that allows us to integrate the algorithms within a complete robotic system ready to be used in real world applications. As can be seen in Figure 2, in each robot, the task allocation algorithm has been integrated with a path planning algorithm and the execution of the tasks are within a behavior architecture that combines a path following algorithm with obstacle avoidance.

As was commented in Section I, one of the main objectives of this paper is to compare the effect of the path planning algorithm on the efficiency of the task allocation algorithms. For that reason, two of the most popular path planning algorithms have been implemented: A^* algorithm [14] and RRTs (Rapidly-exploring Random Trees) [12]. The fist algorithm is based on a heuristic estimator to find the optimal solution faster than general search algorithms such as breadth-first or depth-first search. Even so, for robotic applications, the A^* algorithm still requires a significant amount of processing power. RRTs is also a search algorithm that has a random nature and the quality of the solution cannot be determined a priori, but it is much faster than A^* especially for large state spaces. This algorithm works like a search tree that starts from an initial state and is expanded by performing incremental motions towards the direction of random points. The main difference between this algorithm and a random walk is that the latter suffers from a bias towards places already visited, while RRTs works in the opposite manner by being biased towards places not yet visited.

For navigation, we have selected the DAMN architecture [16] to combine the obstacle avoidance and path follower algorithms. This architecture was designed to combine different behaviors, specially, for mobile robots in unknown and dynamic enviroments. Each of the behaviors votes for a set of possible actuators values satisfying its objectives. Then, an arbitrer combines those votes and generates actions which reflects the behaviors objectives and priorities. Regarding the behaviors, a laser scanner was used as the sensor for the obstacle avoidance and the Pure Pursuit algorithm [15] has been used as the path follower. The Pure Pursuit algorithm geometrically determines the curvature that will drive the vehicle to a chosen path point defined as one lookahead distance from the current position of the robot. Finally, we have used Player/Gazebo [6] to simulate the environment and the robots (see Figure 3).

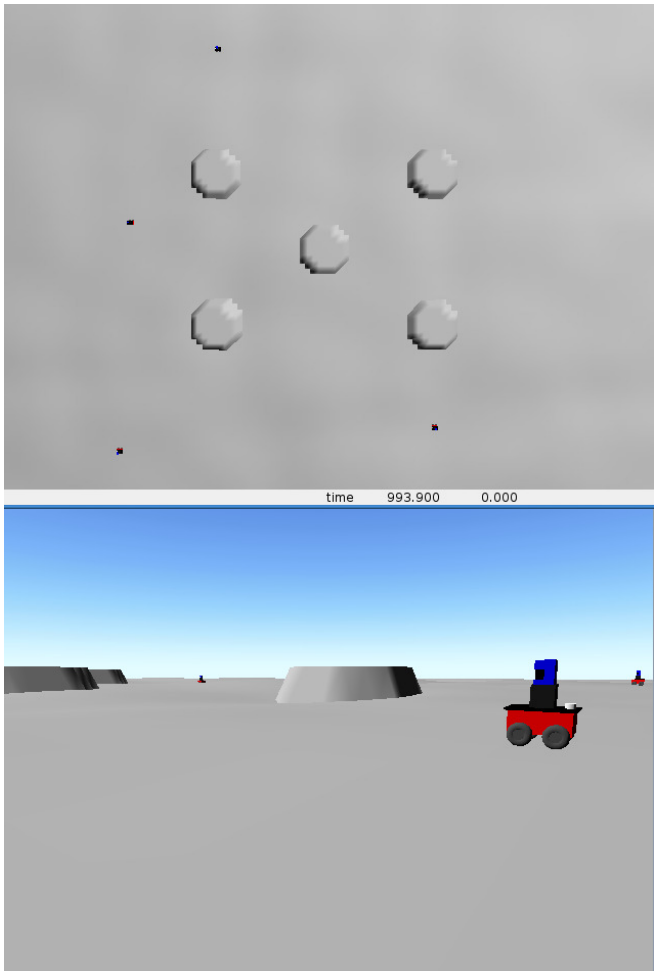

Fig. 3. Snapshot of the simulator Player/Gazebo in a realistic scenario that simulates an Arctic terrain with obstacles. The upper image represents a bird's-eye view of the simulated world. The bottom image is a close view of one of the robots.

IV. SIMULATIONS AND DISCUSSION

A multi-robot simulator has been used to test the decentralized algorithms presented in this paper. This simulator is based on an architecture designed for heterogeneous robots [13] and divided into three layers. The highest layer is independent from the type of robot and is the one aware of the existence of other robots. Thus, the task allocation algorithm is implemented in this layer and can be used, without modification, in both simulations and real robots. Moreover, the communication among robots is based on IP, so it can also be used as an interprocess communication method for simulations. The other two layers are used to execute the different tasks allocated to the robot and make easier the creation of new algorithms by using a modular and component-based architecture.

A. Ideal simulations

In our first set of tests, we have simulated our algorithms in a simple simulator that does not consider most of the real-world concerns. The different algorithms have been tested using initial positions of the robots and formations calculated at random in a virtual world of 1000x1000 meters without obstacles. The simulations have been accomplished using a variety of cases in which the number of robots and tasks ranged from 2 up to 20, and for every case one hundred simulations were run. These results are shown in Table I where, in each cell, the mean of the global cost and the error in percentage in comparison with the optimal solution are presented. The optimal solution has been calculated using the Hungarian method [11]. In order to show the results clearly, only the error in percentage is shown in Figure 4. It can be observed that the RTMA algorithm obtains better results than the BS algorithm, although both algorithms present efficient results up to 8 robots and tasks where the largest error is less than 10%. For more than 8 robots, only the RTMA algorithm obtained good results, with a maximum error of 5.98% in the case of 20 robots. As can be seen in Figure 4, the error with the optimal solution is bounded by a linear function for all the algorithms with the number of robots and tasks. However, the RTMA algorithm is the one with lowest slope. It is also important to point out that for 2 robots and tasks

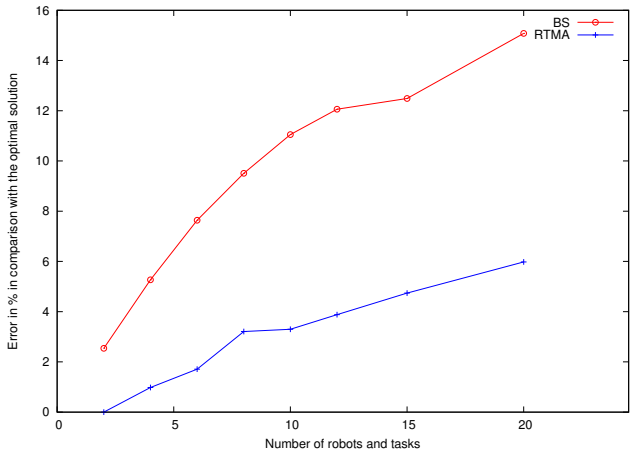

Fig. 4. Error in percentage in comparison with the optimal solution for the two types of algorithms and calculating the inital positions of the robots and the points of the formations at random over 100 simulations. The RTMA algorithm obtains better results than the BS algorithm for all the cases.

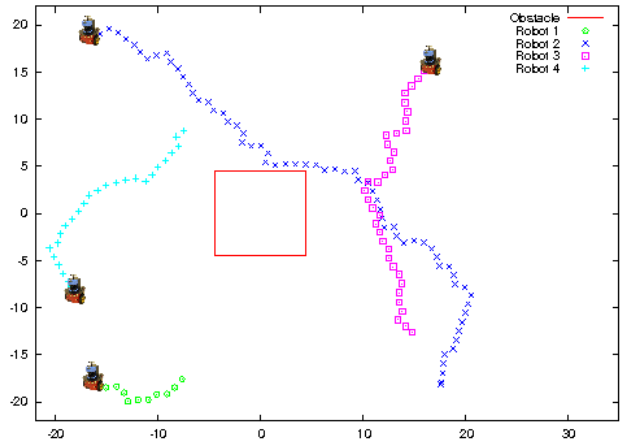

Fig. 5. Scenario with one obstacle (9mx9m) in the middle. The paths show the solution of one of the random missions obtained using the BS task allocation with the RRTs algorithm.

Tasks & Robots	***BS***	***RTMA***	***Optimum***
2	909.35 (2.54%)	886.84 (0.0%)	886.84
4	1473.52 (5.27%)	1413.45 (0.98%)	1399.73
6	2020.13 (7.64%)	1908.85 (1.71%)	1876.77
8	2443.57 (9.51%)	2302.90 (3.21%)	2231.27
10	2865.81 (11.05%)	2666.06 (3.30%)	2580.65
12	3233.25 (12.06%)	2997.34 (3.88%)	2885.35
15	3749.97 (12.49%)	3491.44 (4.74%)	3333.55
20	4639.68 (15.08%)	4272.99 (5.98%)	4031.69

TABLE I

RESULTS COMPUTED FOR FORMATIONS WITH DIFFERENT NUMBER OF ROBOTS AND TASKS OVER 100 SIMULATIONS PER EACH CASE. IN EACH CELL THE MEAN OF THE GLOBAL COST AND THE ERROR IN PERCENTAGE WITH THE OPTIMAL SOLUTION ARE PRESENTED.

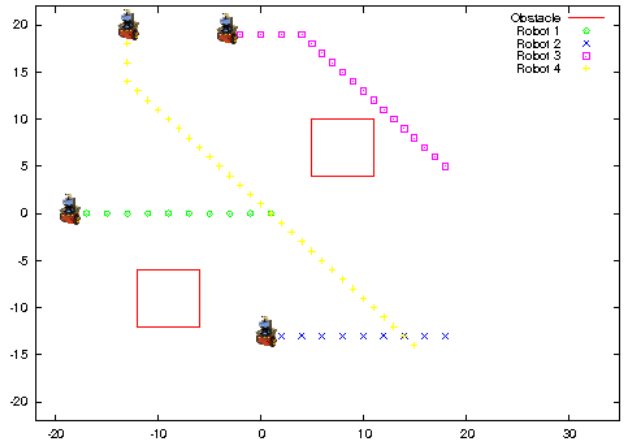

Fig. 6. Scenario with two obstacles (6mx6m each). The paths show the solution of one of the random missions obtained using the BS task allocation with the A^* algorithm.

the RTMA algorithm always obtains the optimal solution.

B. Realistic simulations

The application that we have chosen to simulate our complete robotic system is a science data collection scenario. The main idea is that scientists can use a team of robots to collect data from a hazardous environment, such as the Arctic. In order to facilitate the interface with the robots, scientists just need to point at the areas that they consider interesting from a satellite image as is illustrated in the upper image of Figure 3. The robots, using the commented task allocation algorithms, will divide the different areas and navigate autonomously towards them.

We have used three different scenarios to test our complete robotic system for this type of application. The first scenario (see Figure 5) only has one obstacle in the middle. The second scenario considers two smaller obstacles as can be observed in Figure 6. The last scenario considers five small obstacles (see Figure 7). Also, Figures 5, 6 and 7 show the solution obtained using the BS algorithm and the path followed by the robots and calculated using the RRTs and A^* algorithms. Due to the complexity of these simulations, only 10 simulations have been run per case where the position of the robots and tasks have been calculated at random avoiding the areas considered obstacles in a 40mx40m world.

We first ran our simulations using the A^* for path planning. The results obtained from these simulations are showed in Table II where each cell represents the mean of the global cost over 10 missions, i.e., the sum of the distance traveled by all the robots. It can be seen that the RTMA algorithm still obtains better results than the BS algorithm when it is integrated in a complete robotic system. It can be observed that the results obtained with the complete system are worse, in comparison with the optimal solution, than the results obtained in the previous section (see Table I)

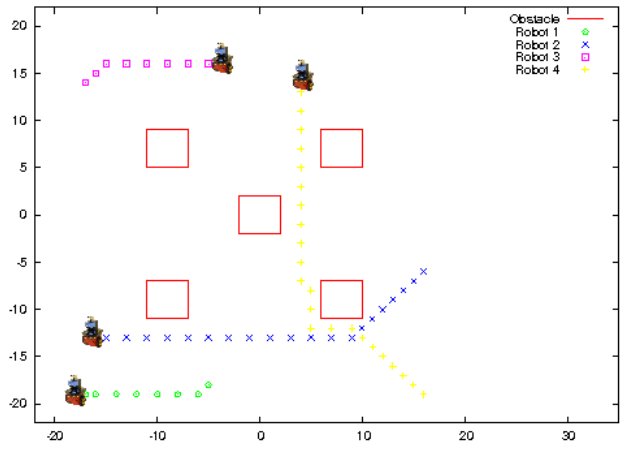

Fig. 7. Scenario with five obstacles (4mx4m each). The paths show the solution of one of the random missions obtained using the BS task allocation with the A^* algorithm.

since a more realistic scenario has been considered. Also, the improvements obtained with the RTMA, in comparison with the BS algorithm, is larger than in the previous results. Therefore, the consideration of a realistic scenario reduces the efficiency of the distributed algorithms in comparison with the optimal solution, but also increases the differences between the algorithms and the effect of the improvements introduced in this paper.

The same random missions have been used for the optimal solution and the two task allocation algorithms. The optimal solution has been calculated using the A^* algorithm with the Hungarian method [11], i.e., all the different optimal paths between every robot and task have been calculated using the A^* algorithm, then the distance of all these paths have been used as the values of the cost matrix that represents the task allocation problem as a job assignment problem. Finally, the Hungarian method has been applied to that cost matrix to calculate the optimal assignment.

Next, we tested our task allocation algorithms with the RRTs instead of the A^* algorithm. The results are shown in Table III. Firstly, it can be observed that these results are worse than using the A^* algorithm. Also, the standard deviations are higher than in the previous case. This makes sense since the RRTs algorithm does not ensure any kind of efficiency of the solution. Finally, it can be observed how the differences between the two algorithms are smaller, and therefore, the variability of the solutions obtained with the RRTs algorithm affects negatively the improvements applied in the RTMA algorithm.

In summary, it has been shown that A^* algorithm obtains better results for our task allocation algorithms. However, the RRTs algorithm should not be completely discarded since A^* might be too slow to be applied in some real-world scenarios, specially, when robots present differential constraints (nonholonomic robots).

Tasks & Robots	*Number of obstacles*	*BS*	*RTMA*	*Optimum*
4	1	88.85 (22.44) 24.84%	82.28 (27.55) 15.61%	71.17
4	2	79.38 (29.46) 31.44%	71.25 (30.31) 17.98%	60.39
4	5	87.02 (17.95) 25.88%	76.79 (14.51) 11.08%	69.13
6	1	118.47 (23.82) 39.24%	101.63 (38.20) 19.45%	85.08
6	2	105.59 (23.49) 31.80%	93.10 (28.22) 16.21%	80.11
6	5	117.57 (38.44) 57.30%	88.80 (20.82) 18.81%	74.74
8	1	150.44 (41.13) 31.41%	140.67 (32.41) 22.88%	114.48
8	2	117.24 (57.65) 38.25%	99.21 (40.08) 16.99%	84.80
8	5	147.24 (49.76) 45.58%	124.00 (38.18) 22.60%	101.14

TABLE II

RESULTS COMPUTED FOR FORMATIONS WITH DIFFERENT NUMBER OF ROBOTS, TASKS AND OBSTACLES OVER 10 SIMULATIONS PER EACH CASE USING THE A^* ALGORITHM. EACH CELL REPRESENTS THE MEAN OF THE GLOBAL COST, THE STANDARD DEVIATION WITHIN BRACKETS AND THE ERROR IN PERCENTAGE IN COMPARISON WITH THE OPTIMAL SOLUTION. THE OBSTACLES ARE DISTRIBUTED AS CAN BE SEEN IN FIGURES 5, 6 AND 7.

V. CONCLUSIONS AND FUTURE WORK

Two different task allocation algorithms that solve the Initial Formation Problem have been described. The first one (BS) based on a market approach is the simplest algorithm but obtains the worst results in all the cases. The other algorithm (RTMA) uses the mean of the costs (considering all the tasks associated to a robot and all the robots for a specific task) in order to increase the information about the whole system and improve the results, but always keeping the distributed computation of the algorithm.

These algorithms have been integrated in a robotic system where the cost of the waypoint tasks are calculated using a path planning algorithm. The execution of the tasks is implemented using a behavior approach where obstacle avoidance and path following are combined using the DAMN architecture.

Different types of simulations have been executed. Firstly, both algorithms have been simulated in a simple simulator where no obstacles have been considered and the cost of the waypoint tasks are just the euclidean distance. On the other hand, these algorithms have been simulated using a realistic simulator with static obstacles and using two different path planning algorithms (A^* and RRTs). The results from all

Tasks & Robots	*Number of obstacles*	*BS*	*RTMA*	*Optimum*
4	1	96.60 (28.20) 35.73%	92.56 (29.88) 30.05%	71.17
4	2	83.43 (30.74) 38.15%	80.70 (31.89) 33.63%	60.39
4	5	98.18 (22.00) 42.02%	89.37 (17.51) 29.28%	69.13
6	1	123.40 (41.68) 45.04%	115.29 (43.01) 35.51%	85.08
6	2	111.98 (36.73) 39.78%	110.60 (31.75) 38.06%	80.11
6	5	115.61 (35.03) 54.68%	106.48 (27.26) 42.47%	74.74
8	1	165.91 (39.29) 44.93%	149.14 (35.99) 30.28%	114.48
8	2	120.73 (43.25) 42.37%	118.21 (40.09) 39.40%	84.80
8	5	159.89 (48.97) 58.09%	143.34 (34.52) 41.72%	101.14

TABLE III

RESULTS COMPUTED FOR FORMATIONS WITH DIFFERENT NUMBER OF ROBOTS, TASKS AND OBSTACLES OVER 10 SIMULATIONS PER EACH CASE USING THE RRTS ALGORITHM. EACH CELL REPRESENTS THE MEAN OF THE GLOBAL COST, THE STANDARD DEVIATION WITHIN BRACKETS AND THE ERROR IN PERCENTAGE IN COMPARISON WITH THE OPTIMAL SOLUTION. THE OBSTACLES ARE DISTRIBUTED AS CAN BE SEEN IN FIGURES 5, 6 AND 7.

these simulations show that the implementation of a task allocation algorithm, within a complete robotic system, gets worse results in comparison with the optimal solution than in a idealistic simulator. Also, it has been observed that the selection of the path planning algorithm affects the efficiency of the task allocation algorithm and it could reduce the effect of the improvements initially designed to obtain better solutions, as happens in the RTMA algorithm when it is used together with the RRTs algorithm.

Future work includes the implementation of these task allocation algorithms with real robots and the proof that the same propertities are still obtained in a real implementation. Also, we will seek to study the effect of mobile obstacles on the efficiency of the task allocation algorithms.

ACKNOWLEDGMENTS

This work supports research in Reconfigurable Sensor Networks for the National Aeronautics and Space Administration, Earth Science Technology Office. The first author would like to thank the Fulbright commission in Spain for providing funding through a Fulbright scholarship.

REFERENCES

[1] S. C. Botelho and R. Alami. M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, 1999.

[2] B Brumitt and A. Stenz. GRAMMPS: A generalized mission planner for multiple mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1998.

[3] M. B. Dias and A. Stenz. Opportunistic optimization for market-based multirobot control. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2714–2720, Lausanne, Switzwerland, 2002.

[4] M.B. Dias. *TraderBots: A New Paradigm for Robust and Efficient MultiRobot Coordination in Dynamic Environments*. PhD thesis, Carnegie Mellon University, 2004.

[5] M.B. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE Special Issue on Multirobot Coordination*, 94(7), 2006.

[6] B. Gerkey, R. T. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th International Conference on Advanced Robotics (ICAR 2003)*, pages 317–323, Coimbra, Portugal, 2003.

[7] B.P. Gerkey and M.J. Matarić. Murdoch: Publish/subscribe task allocation for heterogeneous agents. In *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 203–204, Barcelona, Spain, 2000.

[8] A. Howard and A. Viguria. Controlled reconfiguration of robotic mobile sensor networks using distributed allocation formalisms. In *NASA Science Technology Conference (NSTC 2007)*, Maryland, USA, 2007.

[9] X. Hu and M. Egerstedt. Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, 17(6):947–951, December 2001.

[10] M. Ji and M. Egerstedt. Role-assignment in multi-agent coordination. *International Journal of Assistive Robotics and Mechatronics*, 7(1):32–40, 2006.

[11] H.W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly 2*, pages 83–97, 1955.

[12] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(5):378–400, 2001.

[13] I. Maza, A. Viguria, and A. Ollero. Networked aerial-ground robot system with distributed task allocation for disaster management. In *Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics*, 2006.

[14] N. Nilson. *Problem-Solving Methods in Artificial Intelligence*. McGraw-Hill, 1971.

[15] A. Ollero and O. Amidi. Predictive path tracking of mobile robots. In *Proceedings of 5th International Conference on Advanced Robotics, Robots in Unstructured Environments (ICAR '91)*, volume 2, pages 1081–1086, 1991.

[16] J. K. Rosenblatt. DAMN: a distributed architecture for mobile navigation. *Journal of Experimentation and Theoretical Artificial Intelligence*, 9(2):339–360, 1997.

[17] G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, 29(12), 1980.

[18] R. Zlot, A. Stenz, M. B. Dias, and S. Thayer. Multi-robot exploration controller by a market economy. In *Proceedings of the IEEE International Workshop on Intelligent Robotics and Systems (IROS)*, pages 3016–3023, 2002.