Space D*: A Path-Planning Algorithm for Multiple Robots in Unknown Environments

Article *in* Journal of the Brazilian Computer Society · November 2012

DOI: 10.1007/s13173-012-0071-9

Luan Silveira
Universidade Federal do Rio Grande (FURG)
15 PUBLICATIONS **26** CITATIONS
SEE PROFILE

Silvia Silva da Costa Botelho
Universidade Federal do Rio Grande (FURG)
218 PUBLICATIONS **664** CITATIONS
SEE PROFILE

Paulo Drews-Jr
Universidade Federal do Rio Grande (FURG)
87 PUBLICATIONS **296** CITATIONS
SEE PROFILE

Nelson Duarte Filho
Universidade Federal do Rio Grande (FURG)
45 PUBLICATIONS **53** CITATIONS
SEE PROFILE

Noname manuscript No.
(will be inserted by the editor)

Space D*

A Path-Planning Algorithm for Multiple Robots in Unknown Environments

Luan Silveira · Renan Q. Maffei · Silvia S. C. Botelho · Paulo L. Drews Jr. · Alessandro de L. Bicho · Nelson L. Duarte Filho

Received: date / Accepted: date

Abstract This paper describes a new method of path planning for multiple robots in unknown environments. The method, called Space D*, is based on two algorithms: the D*, which is an incremental graph search algorithm, and the Space Colonization algorithm, previously used to simulate crowd behaviors. The path planning is achieved through the exchange of information between the robots. So decentralized, each robot performs its path planning, which provide a obstacle-free path with the least number of robots around. The major contribution of the proposed method is that it generates paths in spacious environments facilitating the control of robots, and thus presenting itself in a viable way for using in areas populated with multiple robots. The results obtained validate the approach and show the advantages in comparison to use only D* method.

Keywords Path planning · Multi-robot systems · Collision avoidance · Space Colonization algorithm · D* Lite algorithm

1 Introduction

Planning collision-free motions for autonomous robots located in environments with obstacles is one of the main problems of robotics [17] [11] [18].

One area of path planning which has been widely studied is planning in environments with multiple independent robots. For this type of problem, two approaches can be made: centralized and decoupled. In the centralized approach, planning is done by considering a single, large robot made up of several segments (every robot's body), located within a large configuration space (corresponding to the union of the configuration spaces of the robots) where each segment has a desired final configuration (the goal of each robot) [4] [25]. The problem with this approach is that the composed configuration space grows exponentially with the increase in robots' number, which becomes highly infeasible.

In contrast, the decoupled approach, though incomplete, reduces the problem to produce a plan for each individual robot and then adjust them [14] [19]. Furthermore, viable alternatives are distributed decoupled techniques, which take in account the usability of distributed processing techniques to implement such approaches. In these techniques, each robot plans its path based on local knowledge and in interactions with other robots. The main difference between distributed approaches and only-decoupled techniques is that the step of adjusting the path must be distributed and done in real time. Distributed techniques are more robust, because they better accept the fails and uncertainties in the individual robots performances; in other words, a single defective robot does not interrupt the functioning of the whole.

The level of knowledge about the free configuration space regarding obstacle and robot presence is another issue associated with the path planning. A navigation strategy used when there is no accurate knowledge of the environment is called "sensor-based path planning" [8]. In this kind of technique, the shortest path is planned based on the current configuration of the en-

L. Silveira, R. Q. Maffei, S. S. C. Botelho, P. L. Drews Jr., A. L. Bicho and N. L. Duarte Filho
Centro de Ciências Computacionais – C3
Universidade Federal do Rio Grande – FURG
Rio Grande, RS, Brazil
Tel.: +55-53-3233 6875
Fax.: +55-53-3233 6652
E-mail: silviacb@furg.br

vironment and, as soon as new obstacles are detected, new paths must be planned.

One of the most popular algorithms that handle this type of problem is the D* algorithm (Focused Dynamic A*) [29], which adapts the optimality of A* for dynamical environments, mixing incremental searches with heuristics, and achieving significant gains over repeated executions of A*.

Several works have been proposed based on D*. For example, D* Lite [16] implements the same strategy of D*, but in a simplified manner and with an efficiency equal or greater than the D*. In [10], it was proposed an extension to the D* and D* Lite, using linear interpolation to produce smoother paths, bypassing the issue of limited possibilities of transition between cells. Another extension of the D* Lite was proposed in [20], where the algorithm was adapted for systems with limited time. In this case, as in [2], the planning is done incrementally and further refined in the available time.

Another important issue is the existence of a environment populated by dynamic elements, *e.g.* fleets of mobile robots. The possibility of collision between robots increases as the number of robots grows. Planning algorithms can take into account these risks in advance, leading to plans to prevent collisions by searching for areas with lower density of occupation.

The current work proposes an extension of the D* algorithm for multi-robots environments, aiming to provide a collaborative path-planning approach for multi-robots world populated. The algorithm, called Space D*, combines the D* algorithm with the Space Colonization algorithm, originally designed to simulate the plant modeling [28], and later adapted for the simulation of crowd behaviors [3]. The main feature of the Space Colonization algorithm is the preference shift by spaces, to avoid the collision risk given the uncertainty and scalability of mobile and fixed obstacles. The present work is an extension of the work previously published in conference proceedings [21], where the method is better explored and new results using real robots are showed.

In crowded environments, the difficulty of having a reliable measure of the proximity between the robot and obstacles must be addressed. Methods that simply seek the shortest path tends to get closer to obstacles, requiring precise control of the trajectory, for example by reducing the speed of the robots, and therefore increasing the total execution time (in spite of obtaining lower total distance traveled). Moreover, some methods that prefer free environments, such as potential fields, do not guarantee find the paths (fall into local minimal), unlike the Space D*, which inherits this characteristic from D*. Our approach is suitable for use in real environments by focusing on the creation of paths through environments with more free space.

This article is organized in five sections. Section 2 presents some related works about path planning of multi-robot in unknown environments. In Section 3, we show techniques that serve as the basis for the developed method. Section 4 presents the path planning algorithm proposed in this paper, dubbed Space D*. Section 5 provides the tests performed on the developed system, as well as the validation of the method. Finally, Section 6 presents the conclusions and suggestions for future work.

2 Related Works

Many studies on path planning of multiple robots are based on environments that must be completely known [9] [1]. Thus, the application of techniques such as prioritized motion planning and fixed-path coordination in unknown environments can be achieved, as long as together with some approach to treatment of uncertainty.

Moreover, the algorithms for path planning in unknown environments, such as [29] [16] [10], are designed to independents robots. However, they can be applied to problems with multiple robots, considering the other robots in the scenario as dynamic obstacles.

There are some works that exploit the presence of multiple robots with the ability to communicate and develop techniques for collaborative path planning dealing with uncertainty. Most of the works of collaborative path planning for multiple robots located in unknown environments focused on the problem of exploration of the environment.

In [27], the environment is separated into stripes that are explored by teams of robots and, within each team, only one robot moves at a time, reducing odometry errors. In [22], robots explore the environment randomly and exchange information about obstacles when they meet. In [32], robots seek for the nearest unknown areas according to their incremental maps. Although, there is not coordination to prevent that two robots do the same path (improvement proposed in [5]). Later, in [6], it was proposed an algorithm based on cost and utility functions to arrive at a particular uncharted location. A collaborative exploration is the focus of works like in [12] and [31], among others.

Nevertheless, there are some works focused on the path planning itself. In [26], using navigation functions, the robots remain close to each other in order to exchange information, while they moving in toward their goals. In [7], robots are guided to a common destination by a specific robot, while the others are responsible for collecting the information from the environment.

In [23], it is proposed an algorithm based on Probabilistic Roadmaps (PRM), where the robots have different priorities (required to prevent collisions), and they are able to exchange information about their paths. In [24], social rules are used to perform local planning in robots using a decentralized way, avoiding the occurrence of collisions. In [15], robots with common goals exchange information on the map during crossing, it prevents that areas being revisited.

Our work seeks to take in account the preference for navigation in non-populated spaces (appropriated for dynamic unknown environments), ensuring a solution path (if there is) and a collaborative mapping.

3 Underlying Foundations

As mentioned earlier, the Space D* is based on two algorithms that are discussed briefly below: the first is the D*, widely used for path planning in unknown environments, and the Space Colonization algorithm, which was developed for modeling plant growth [28] and used for the simulation of crowd behaviors [3].

3.1 D* Algorithm

As presented, the D* algorithm [29] (replaced now by D* Lite [16]) is one of the most used and efficient approaches to the problem of path planning in unknown environments. In fact, after its publication, the D* Lite became used in place of D* for most works, including the present work.

The basic behavior of the algorithm consists in recalculate the shortest path to the destination whenever the cost of some cell varies (an obstacle or free space is found). This can be seen in Figure 1, where the robot's current position is the node marked with a spot, the destination is the node marked with an "x" and the computed path are the hatched nodes. The nodes in black and white are those with the costs already observed, representing, respectively, free spaces and obstacles.

The operation of D* Lite is inspired by the A*, being that for each node s of the graph (indicating a position on the environment) is calculated the distance to reach the goal (s_{goal}), and whenever the cost $c(s, s')$ to go from s to a neighbor node s' changes (for example, if an obstacle is detected), it should be made the update of the distances of nodes affected by this change.

The algorithm uses two estimates of distance to the goal starting from a node s, called $g(s)$ and $rhs(s)$.The estimate $rhs(s)$ is potentially more accurate than $g(s)$, once it is based on the values of $g(s)$ of the successors nodes of s. The basic principle of the algorithm is keeping the two values consistent for all nodes, which means that all estimates of distances are correct.

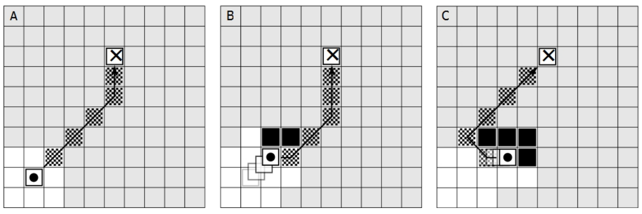


Fig. 1: Operation of D*/D* Lite. a) The robot starts calculating a shortest path from their position (spot) to the goal (cross). b) While move toward the goal, it senses the environment and detect obstacles. In the case of blocked paths, a new path need to be calculated. c) This process repeats until the goal is reached.

When there is any inconsistency between a pair of estimates, derived from the change in the cost of a node, the algorithm must update the estimates of every node affected by this change. However, to avoid having to go through all nodes in the graph, the D* Lite uses a heuristic to create keys that indicate the priority of analysis of each node. With this heuristic, the algorithm computes only the nodes that seem relevant to find the shortest path.

3.2 Space Colonization Algorithm

Based on studies of biology, that attributed the development of veins in leaves to a hormone called auxin found in plants, the Space Colonization algorithm was developed in [28], to simulate the growth of the venation pattern in relation to the distribution of auxins in the leaves.

The algorithm is basically composed of three processes. The first is the growth of the leaf. Alongside, it happen the other two: the development of the venation pattern, which is directly influenced by the distribution of the sources of auxins in the leaf (representing free spaces), and the generation of new auxins, which in turn is affected by the development of veins.

The vein growth begins by associating each node of auxin to the nearest vein node. Then, the nodes are expanded in the direction of the associated auxins. As the pattern grows and approaches the auxins, these are being removed, since the space that they represent is no longer free. The leaf blade grows and new auxins are randomly generated, uniformly distributed through free space. The cycle begins again with the auxins-nodes association, and continues until the leaf reaches its maximum size. As can be seen, the algorithm is characterized as a competition of vein nodes for auxins, representatives of free spaces, since the more auxins a node can be associated, more space he has to expand.

In [3], the Space Colonization algorithm was adapted to simulate crowd behaviors. The concepts of vein nodes and auxins were transformed into agents and markers of free spaces. In such case, the Space Colonization algorithm was used to simulate the competition of agents for free space on environment.

Also in [3], some modifications on the algorithm were proposed. Firstly, the markers have become persistent, not being "consumed" by the agents, but instead, allocated when an agent comes up and released when it leaves. Another change is that only the markers within a certain region around an agent can influence it (previously, any auxin could influence any leaf node, provided it was the closest one). Moreover, the direction of motion is guided by the intention of reaching a specific destination, not only by large availability of markers. And last one is that the speed of movement of the agents varies according the free space (in the original algorithm, the speed was uniform).

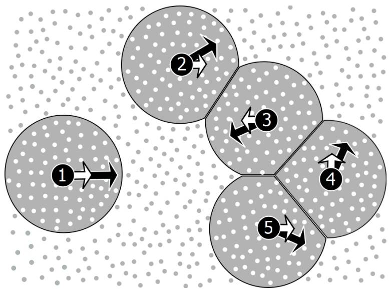


Fig. 2: Space Colonization algorithm for simulation of crowd behaviors. White Arrows indicates the intention of movement. Based on the free space that the agent could allocate (hatched areas), a new direction of movement is calculated (black arrows).

Figure 2 shows the example of the algorithm execution. The agents are numbered black circles with white arrows indicating the direction of their goals. Larger areas are the regions of markers that each agent was able to allocate. As can be observed, the restrictions in the allocated areas directly influence the movement vector of each agent, represented by the black arrow. Each agent seeks to move toward its goal, but avoiding the proximity to other agents or obstacles. The Space Colonization algorithm by itself does not solve neither the problem of path planning in environments with obstacles nor the local minimum issues, but its feature of search for free spaces has motivated to merging it with the D* Lite.

4 Space D*

The algorithm developed in this work for path planning of multiple robots in unknown environments, named Space D*, relates the D* algorithm with the Space Colonization algorithm. The configuration space is represented by a grid formed by markers allocated by the nearest *owner robots'* during the path planning. Markers have a cost which describes the level of difficulty that the robot has to reach the goal through this intersection. The owners of each markers and the cost associated to them changes over time. The operation of the method is, briefly, the following (for each robot):

– The environment is represented by a grid of markers, that indicates the costs for a robot to cross it and to reach a goal (obstacles have the highest cost of all). At first, the map is empty (zero cost) and as new obstacles are detected, the map is updated;
– The D* Lite algorithm is applied by each robot on the respective set of markers, updating the cost of this markers(distance to its goal);
– The robot navigation is done using a modification upon the Space Colonization algorithm. The robot allocates all near markers, within its reach, that indicates free-spaces. With the distances to the goal of these markers, calculated by the D* Algorithm, it calculates its direction of movement, which will always be defined within the allocated zone;
– When the robots meet each other, the information about their maps are exchanged, allowing to update the costs of markers. In addition, the algorithm tries to avoid collisions between them, since each robot only allocates the markers that are closer to it than to other robots;

A major contribution of the proposed method is the inclusion in D* of the preference for navigating in less populated environments, which improves the handling of unknown and dynamic environments. In its original definition, D* generates paths that tend to be very close to obstacles in order to reduce the total distance traveled, but this ends up requiring a very precise control over the movement of robots.

Another advantage of the proposed method is the exchange of information between the robots, focus of related works, such as [15]. Since each robot obtained the map of the environment where it traveled (to perform the calculation of the distance to the goal), information about the presence of obstacles can be exchanged when robots cross each other. This raises the possibility of avoiding paths that are blocked and that only eventually the robot would notice.

4.1 Operation of Space D*

The Algorithm 1 show the operation of Space D*.

Algorithm 1: Space-D*

```
begin
    while Did not reached the goal do
        Maps the environment.
        if Detect other robots then
            Exchange information.
        end if
        Update markers cost.
        Run D* Lite.
        Allocates the nearest markers.
        Calculates the direction of movement.
    end while
end
```

The two initial stages of the process, related to the perceptual system of the robots, are the mapping of the environment and the detection of other robots. In the environment mapping, each robot senses the presence of obstacles in order to define the costs of the markers associated with each position. As for the detection of robots, it was defined that each robot has a unique identifier, which makes that when a robot moves close enough to another, he can realize that the obstacle is actually a robot and also which robot.

The third stage is the exchange of information, which follows the detection of robots. When two or more robots meet, both stop for a moment and exchange information corresponding to their descriptions of the map. It is noteworthy that in the current implementation, all obstacles are static (except for the robots themselves), so the upgrade of the markers costs from the exchange of information consists of verifying if, in the received map, there are new obstacles that in the current map has not been discovered yet.

The fourth stage is the update of the markers costs, resulting from the steps of environment mapping and information exchange. Later, are computed the distances to the goal starting from each marker of interest of the robot (the $g(s)$ estimative from D* algorithm). The processing performed in this step is, in fact, calculating the shortest path according to the D* Lite.

In the sixth stage, all markers contained in the environment are associated to their nearest robot, provided they are within the reach of the robot. The difference from the simulation of crowd behaviors [3] is that in the proposed method this step is decentralized, as it is up to each robot to allocate all the markers located within its reach, considering only those that are closer to themselves than to other robots.

A restriction lies on the size of the radius of the allocated region. To avoid collisions between robots is necessary that the reach of robot perception sensor always be greater than the radius of the area of allocated markers. Ideally, the sensor reach has to be at least twice the radius of allocation, as this ensures that, in the exact moment when two robots meet, each one is allocating only the markers closest to them.

In the final step, to calculate its direction of motion, each robot uses the distances to the goal of the markers from their area of interest. The calculation of movement vector $\overrightarrow{m}$ is a weighted vector sum of the vectors connecting the robot ($\overrightarrow{r}$ position) to each associated marker ($\overrightarrow{s}position$):"

$$\overrightarrow{m} = \sum_{k=1}^{N} f_k(\overrightarrow{s}_k - \overrightarrow{r}). \qquad (1)$$

Where N is the number of markers in the set $S(r)$. But unlike [3], where the size of each vector was related to the direction of the goal, now the modules of the vectors are based on the distance between marker and goal, so as the smaller the distance, the greater must be the module of vector. The definition of the function f that determines the magnitude of the vector of a marker s associated with the robot r is:

$$f(s) = max_{p \in S(r)}(g(p)) - g(s). \qquad (2)$$

That is, the size of each vector is the subtraction from the largest distance $g(p)$ of a marker belonging to the set $S(r)$ of the robot, by the value of g of each marker.

In Figure 3 can be seen an example of the execution of the Space D *. The position of the robot is the node indicated with a spot, the goal is the node indicated with an "x" and the obstacles are the black nodes. In (a), the demarcated area represents the allocated region of the robot, with the distances to the goal of each marker (in dotted lines is the shortest path calculated by D* Lite). In (b), the vectors are generated in the direction of the markers with the module calculated by $f(s)$, later, with them it obtains the movement vector (larger vector). In (c), after moving in the direction indicated previously, the robot perceives new obstacles and recalculates the distances to the goal. In (d), a new movement vector is generated.

5 Tests and Validation

This section presents a series of experiments in order to validate the proposed method. In contrast with [21], the experiments used in this paper included the use of real robots, it makes possible to verify the robustness of the method in noisy environments. Therefore, the

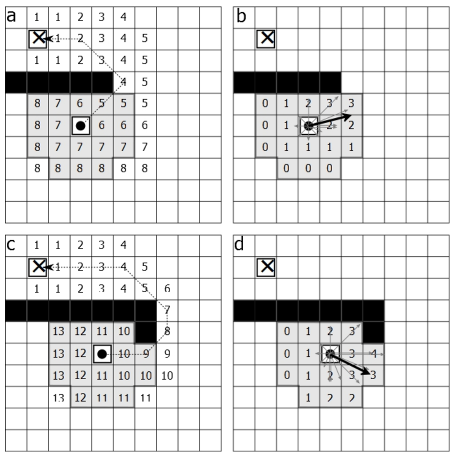


Fig. 3: Example of operation using Space D*. (a) The current map of the robot, showing the obstacles in black and the shortest path from the current position (circle) to the goal position (cross), obtained by the execution of the D* algorithm. (b) The markers allocation area of the robot and the direction of movement (black arrow), obtained after the execution of Space Colonization Algorithm. (c) New obstacles are discovered in the environment, implicating update in the path to the goal. (d) The new direction of movement, based on the new allocation area and shortest path.

algorithm Space D* was validated in simulated environments for multi-robots on the software Player/Stage [13], as well as using real robots controlled using Player software. A detailed description of experimental results is showed in next sections.

5.1 Simulated Tests

The validation using simulated robots was performed by comparing the solutions obtained in the execution of the Space D* with the results of the execution of D* Lite. The simulations use omnidirectional robots with 50cm of radius in one environment with $100m \times 100m$.

Spacious Paths It was observed that, like the D* Lite, the Space D* generates trajectories that always reach the goal without the occurrence of collisions with obstacles, but in addition, as was predicted, it tries to maintain the robot far from any obstacle, either dynamic (other robots) or static (walls).

For the method used for comparison, the direction of movement of robots is the indicated by the shortest path computed by D*. As can be inferred, if there are no errors and inaccuracies in the control of robots, this technique works perfectly. Yet, collisions often occur when paths are generated too close to obstacles, unless the displacement speeds of the robots are reduced in these locations.

Figure 4 shows a comparison between the trajectories generated by the execution of the Space D* and D* on the environment showed in Figure 5. It displays the paths traveled by three robots dispersed randomly. It is possible to see that the trajectories obtained with the Space D* deviates more from obstacles than those obtained by D*, which decreases the need for speed reductions.

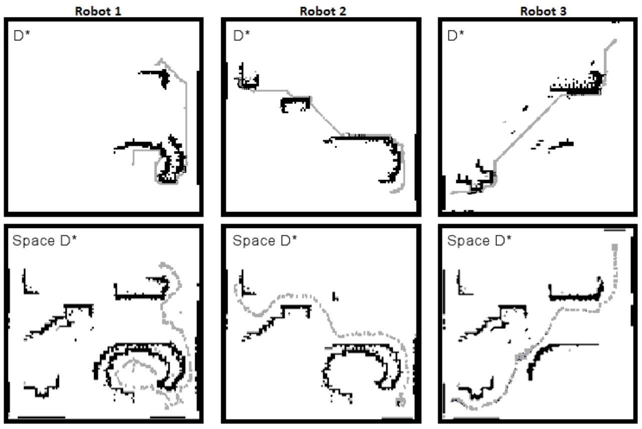


Fig. 4: Comparison between the trajectories was generated by the execution of the Space D* and D* for three robots. The black segments are obstacles and the gray are the trajectory generated by the algorithms. We can observe on the top images, the paths generated by the D* almost touch the obstacles what difficult the control of the robots. In contrast, the Space D* generates loose paths, that keep the robot far from obstacles.

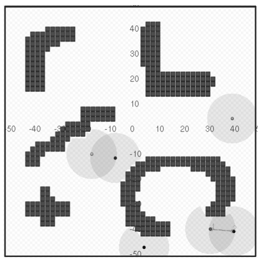


Fig. 5: Map used to compare the D* Lite and the Space D*.

Average Performance: execution time, distance and velocity In Table 1 was compared the average execution times, average total distances and average speeds of robots between the simulations using the Space D* and the simulations using only the D*. In all test cases, the environment was 100x100 meters (and it was used a discretization of 100x100 markers) and the maximum speed of robots was 5 m/s. It was evaluated three test sets (10, 15 and 20 robots), each one with five random configurations of initial and final positions for each robot.

It must be reiterated that in terms of execution time of algorithm, the Space D*, by making a larger processing, is slower than the D* Lite. In fact, using the D* Lite, each simulation cycle was approximately half the cycle time of the Space D*. However, by avoiding the approaching with obstacles, the Space D* allows a refresh rate smaller. In the result obtained with the Space D* the average time simulation was always lower (about 30%), because although the D* obtains smaller trajectories (as can be seen by the average distance), its average speed is considerably lower, once the speeds of the robots are reduced, in the proximity of obstacles. This indicates that the use of Space D* it was more advisable, since yet the D* needs to treat the uncertainty over the configuration space and the control of robots (for instance, reducing the speed of the robots).

Exchanging Information, high scalability and complex environments Following, it was examined if the existence of information exchange between robots contributes significantly to the performance of the Space D*. It was tested with two different maps as shown in Figure 5 (the map used for the previous tests) and Figure 6 with three test sets (10, 15 and 20 robots), each one with five random configurations of initial and final positions for each robot. The results are shown in Table 2. For the first map, the exchange of information did not contributed to an improvement in execution time. Instead, it was proved worse than the method without exchange of information for a sample with 10 robots. Nevertheless, in tests with 20 robots, the method with the exchange of information was a little better. However, for the second map, the method with the exchange of information proved to be far more efficient than the method where there is no exchange of information.

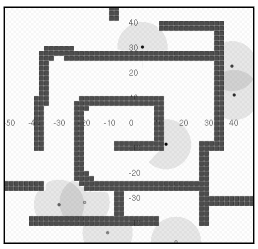


Fig. 6: Second map used in tests.

It can be inferred when obstacles are sparse and easily circumvented, the exchange of information does not make much difference. However, in an environment similar to a maze, the choice of the path to be taken by robot is a critical decision. The more information about the environment the robot has, the greater the certainty in making this decision and the smaller the distance to the goal.

5.2 Real Robots

The robots used in the experiments are three iRobot Create equipped with Hokuyo Laser Scanner URG-04LX-UG01, with 0.36^0 of angular resolution, 180^0 of angular range and $0.5m$ of maximum range distance. The laser scanner is distanced $0.12m$ from the center of the robot. Due to the differential wheels configuration of the robots, for each velocity vector sent to them, firstly we rotated to agree with robot's orientation with the velocity vector and, after, translate with vector modulus.

Moreover, each robot has global localization using a system based on ARToolkiPlus [30] markers imaged by one low cost USB Webcam with 640×480 pixels resolution. This system able we to know the robots pose in each time. Despite this global knowledge, the path planning algorithm was executed in a distributed way. The allocation markers radius is defined in the algorithm as $0.5m$.

Results were obtained using real robots in two different situation. The first situation uses two robots in one simple maze, composed by three walls. Figure 7 shows the real environment with robots, the real path, the initial position and the goal, as well as the obstacles. Figure 7-a illustrates the initial configuration in the experiment, and Figure 7-c shows the final configuration of the experiment. Figure 7-b shows a conflict situation, where the path determined by the algorithm generated the possibility of collision. Due to the Space Colonization algorithm, both robots recalculate the path in order to deviate and achieve the goal. The map acquired by each robot is showed in Figure 8 and 9, respectively. Each of these maps represents approximately the same time instant showed in Figure 7, *i.e.* the initial position, conflict situation and final position. The shaded area in yellow represents the allocation of space made by the Space Colonization Algorithm at that moment. The Robot 1 is illustrated using black color in Figure 7, *i.e.* the robot that start in left position and finished in right position. It spent approximately 44s in its path. Robot 2 is illustrated in blue color and it spent approximately 56s in its path. The difference in the elapsed time by each robot is due the distance between the start

	Using the Space D*					
No. of Robots	Time (s)		Distance (m)		Speed (m/s)	
	Mean	StDev	Mean	StDev	Mean	StDev
10	25.6	7.6	89.74	38.16	3.71	0.82
15	29.7	12.2	88.91	36.66	3.03	0.7
20	33.2	16.8	86.29	35.07	2.69	0.61
	Using only the D*					
No. of Robots	Time (s)		Distance (m)		Speed (m/s)	
	Mean	StDev	Mean	StDev	Mean	StDev
10	33.2	10.3	83.21	29.61	2.57	0.64
15	42.4	16.0	85.68	31.07	2.13	0.58
20	50.9	21.8	86.91	33.48	1.74	0.39

Table 1: Comparison between Space D* and D*.

Map 1

	Method with exchange of information					
No. of Robots	Time (s)		Distance (m)		Speed (m/s)	
	Mean	StDev	Mean	StDev	Mean	StDev
10	25.6	7.6	89.74	38.16	3.71	0.82
15	29.7	12.2	88.91	36.66	3.03	0.7
20	33.2	16.8	86.29	35.07	2.69	0.61
	Method without exchange of information					
No. of Robots	Time (s)		Distance (m)		Speed (m/s)	
	Mean	StDev	Mean	StDev	Mean	StDev
10	23.8	6.5	89.23	38.56	3.91	0.88
15	29.6	11.8	90.86	39.9	3.12	0.69
20	35.4	17.9	92.15	41.08	2.74	0.63

Map 2

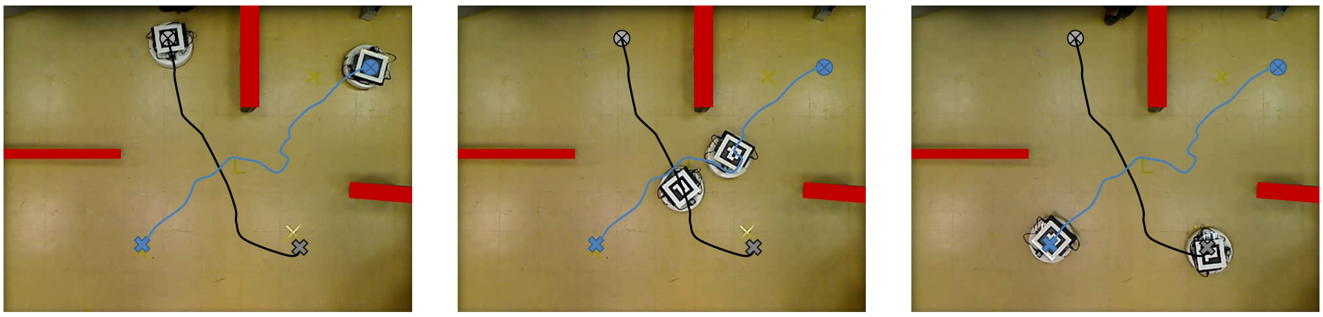
	Method with exchange of information					
No. of Robots	Time (s)		Distance (m)		Speed (m/s)	
	Mean	StDev	Mean	StDev	Mean	StDev
10	37.9	18.3	126.26	63.71	3.43	0.74
15	39.4	19.0	113.05	52.12	2.89	0.66
20	40.3	19.7	99.74	44.48	2.41	0.42
	Method without exchange of information					
No. of Robots	Time (s)		Distance (m)		Speed (m/s)	
	Mean	StDev	Mean	StDev	Mean	StDev
10	42.2	22.1	142.65	71.09	3.51	0.83
15	47.9	24.9	145.33	75.39	3.07	0.7
20	51.8	27.2	146.14	78.95	2.82	0.61

Table 2: Comparison between methods with and without exchange of information, on the map of Fig. 5 (map 1) and on the map of Fig. 6 (map 2).

and goal positions and, mainly, the conflict situation where one robot has captured markers that allowed it to achieve faster the goal.

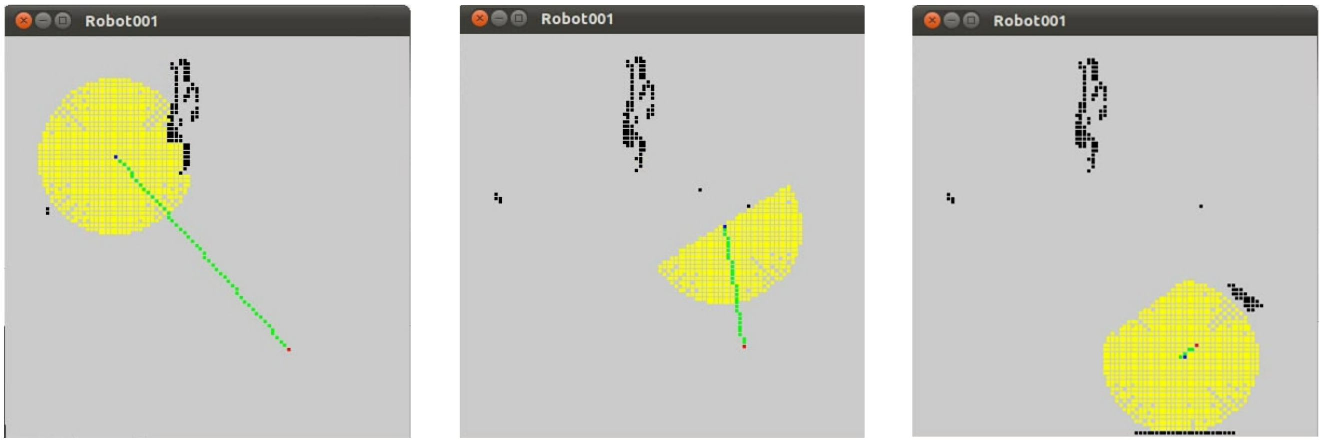
The second experiment obtained using three robots without obstacle. In this case, only the robots represent obstacles. Figure 10 shows the real environment with three robots, the real path, the initial position and the goal. Figure 10-a illustrates the initial configuration in the experiment, and Figure 10-c shows the final configuration of the experiment. Figure 10-b shows a conflict situation, where the path determined by the algorithm generated the possibility of collision. Due to the Space Colonization algorithm, both robots recalculate the path in order to deviate and achieve the goal, as in previous experiment. The Robot 1 is illustrated using black color in Figure 10, *i.e.* the robot that start in the top right. It spent approximately 1 minute in its path. Robot two is illustrated in green color, *i.e.* the robot that start in the bottom left, and it spent approximately 56s in its path. The last robot showed in blue color, *i.e.* the robot that start in the bottom right, it spent approximately 43s in its path. Both the videos, with two and three robots, are available in the research group website.

The information exchange was not tested in real robots due to the limited size of the tested area. This exchange gives more information to the robots, making easier to solve the path planning in this case. Thus, in



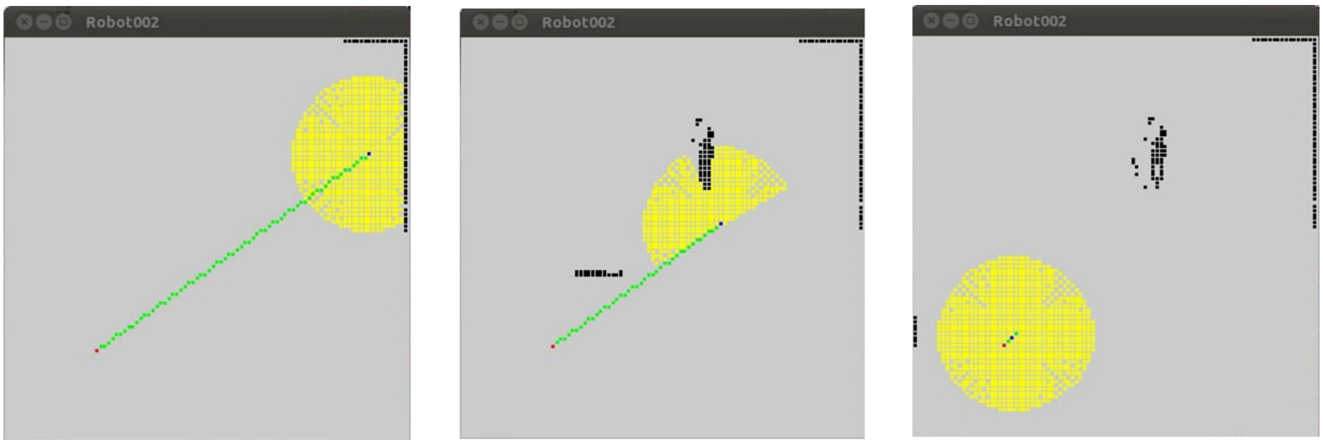
(a) (b) (c)

Fig. 7: Real experiment with two robots, with the real path, the initial position (circle), the goal (cross) and obstacles. a) image in the initial situation; b) conflict situation where the paths of both robots intersect; c) final position.



(a) (b) (c)

Fig. 8: Map of Robot 1 (black in Fig. 7) in: a) initial situation; b) conflict situation, where the paths of both robots intersect; c) final position.



(a) (b) (c)

Fig. 9: Map of Robot 2 (blue in Fig. 7) in: a) initial situation; b) conflict situation, where the paths of both robots intersect; c) final position.

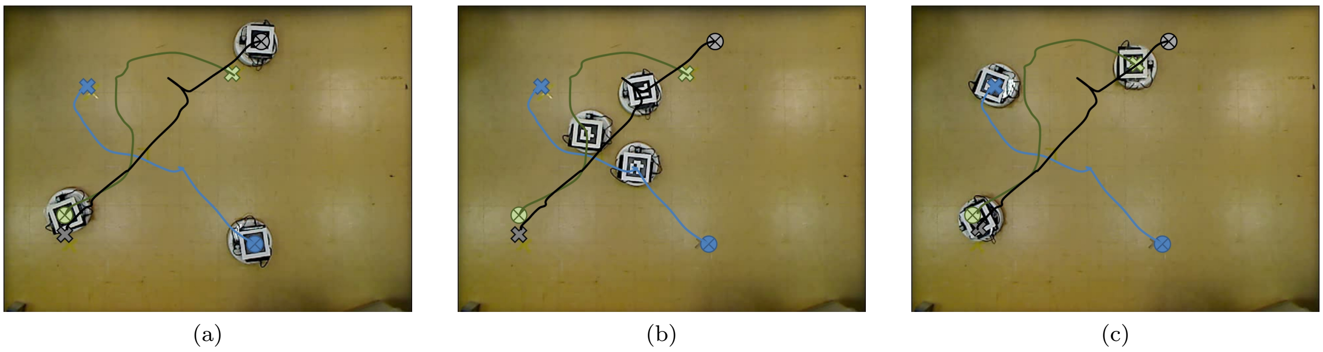

(a) (b) (c)

Fig. 10: Real experiment with three robots, with the real path, the initial position (circle) and the goal (cross). a) image in the initial situation; b) conflict situation where the paths of all robots intersect; c) final position.

all real tests we have disabled the information exchange function.

Notice that the same restrictions that exist in D* are still present at Space D*. A major constraint is the level of granularity associated with space discretization. The Space D* is based on a graph-search algorithm, being costly for large environments.

Another important restriction of the method is that in its current implementation it considers the robots as infinitesimal, that is, their bodies are considered points. For proper functioning of the algorithm, it must be ensure that all robots never leave their regions of allocated markers and, as these regions reduces, their movements also decreased. For real robots, the algorithm ensures that the centroid of the robot will never leave the bounded region, however this can cause problems when the allocated area of a robot is very small.

6 Conclusion

The algorithm Space D* focuses on generating paths over spacious environments, keeping the robot far from dynamic and unknown obstacles (wall, obstacles and other robots). Thus, the method is applicable in realistic situations, where using only the D* does not seem a viable alternative. When compared with other methods of path planning, like Potential Fields, this algorithm has the advantage of not suffer of local minimums, characteristics inherited from D*.

The framework proposed for mapping representation and exchange of information between robots presents gains in the run-time in environments where obstacles form excerpts of dead ends. Real results showed the robustness of the method, where without a good controller and sensory is possible to achieve path planning. Particularly, in Table 2 we can verify that the exchange of information provided better results in complex environments. Once the robot has more information about the environment in which he must move, the greater the chances to decide correctly the path to your goal. It is possible to verify that the greater number of robots in complex environments with exchange of information allow smaller distances. Furthermore, the reduction of the mean velocity of the robots in this context is insignificant compared to the gain obtained in the mean distance travelled by the robots.

The complexity of the algorithm proposed is governed by the execution of D*, that is $O(k\log(k))$, where k is the number of markers in the environment. The complexity of the algorithm to exchange of information increase linearly with the number of robots n, leading a $O(n)$ complexity. Besides that, only a few robots are communicating at each time, because two robots only exchange information when they get close of each other. Therefore, the orders of complexity presented and the restriction of the robots communicating with each other at only short distances allow the scalability of the system.

Since the proposed method is decentralized, it is possible that the algorithm fails in specific situations. For example, if two robots meet in a narrow hallway that requires one or the other to reverse its direction. Using the Space Colonization and D* algorithms, it would be possible that one of the robots determining a reverse direction for a significant distance in order to resolve the conflict. Our method would present the solution described if the diameter of the robots was exactly the width of the corridor, which would be a situation not likely to happen. If there is a small space between the wall of the corridor and the robot, the proposed method will indicate this availability for the other one. Consequently, the robots probably would be stagnant because there was not enough space to move. Therefore, we suggest as future work to solve problems related to the bottleneck effect.

Another future work, there is an idea of associating confidence levels with markers as well to extend the algorithm to work with dynamic obstacles. Another future direction is including in the calculation of markers costs and concentration of robots in areas of environment, using exchange of information. It makes possible to find places with higher concentrations of robots, and consequently avoid them. Moreover, we intend to make more results in a large environment with more robots in order to consolidate our approach.

Acknowledgements This work has been supported by grants from CNPq, FAPERGS, CAPES and FINEP. The authors would like to thank to VerLAB members from DCC-UFMG by collaborate with the real experiments, especially to Mario Campos, Elizabeth da Costa, Gabriel Oliveira, Thiago Sonego and Rafael Colares.

References

1. van den Berg, J., Snoeyink, J., Lin, M., Manocha, D.: Centralized path planning for multiple robots: Optimal decoupling into sequential plans. In: Proceedings of the Robotics: Science and Systems (RSS) (2009)
2. van den Berg, J.P., Ferguson, D., Kuffner, J.: Anytime path planning and replanning in dynamic environments. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 2366–2371 (2006)
3. Bicho, A.L., Rodrigues, R.A., Musse, S.R., Jung, C.R., Paravisi, M., Magalhes, L.P.: Simulating crowds based on a space colonization algorithm. Computers and Graphics **36**(2), 70–79 (2012)
4. Bonert, M., Shu, L.H., Benhabib, B.: Motion planning for multi-robot assembly systems. International Journal of Computer Integrated Manufacturing **12**, 301–310 (2000)
5. Burgard, W., Moors, M., Fox, D., Simmons, R., Thrun, S.: Collaborative multi-robot exploration. In: Proceedings of the IEEE International Conference on Robotics And Automation (ICRA), pp. 476–481 (2000)
6. Burgard, W., Moors, M., Stachniss, C., Schneider, F.E.: Coordinated multi-robot exploration. IEEE Transactions on Robotics **21**, 376–386 (2005)
7. Chen, J., Li, L.R.: Path planning protocol for collaborative multi-robot systems. In: Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, pp. 721–726 (2005)
8. Choset, H., Burdick, J.W.: Sensor-based planning and nonsmooth analysis. In: Proceedings of the International Conference on Robotics and Automation (ICRA), pp. 3034–3041 (1994)
9. Clark, C.M., Rock, S.M., Latombe, J.C.: Dynamic networks for motion planning in multi-robot space systems. In: Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS) (2003)
10. Ferguson, D., Stentz, A.: Field D*: An interpolation-based path planner and replanner. In: Proceedings of the International Symposium on Robotics Research (ISRR), pp. 1926–1931 (2005)
11. Fraichard, T.: Trajectory planning amidst moving obstacles: Path-velocity decomposition revisited. Journal of the Brazilian Computer Science **4**(3) (1998)
12. Franchi, A., Freda, L., Oriolo, G., Vendittelli, M.: The sensor-based random graph method for cooperative robot exploration. IEEE/ASME Transactions on Mechatronics **14**, 163–175 (2009)
13. Gerkey, B.P., Vaughan, R.T., Howard, A.: The Player/Stage project: Tools for multi-robot and distributed sensor systems. In: Proceedings of the 11th International Conference on Advanced Robotics, pp. 317–323 (2003)
14. Guo, Y., Parker, L.E.: A distributed and optimal motion planning approach for multiple mobile robots. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp. 2612–2619 (2002)
15. Jung, J.H., Park, S., Kim, S.L.: Multi-robot path finding with wireless multihop communications. IEEE Communications Magazine **48**, 126–132 (2010)
16. Koenig, S., Likhachev, M.: Improved fast replanning for robot navigation in unknown terrain. In: Proceedings of the International Conference on Robotics and Automation (ICRA), pp. 968–975 (2002)
17. Latombe, J.C.: Robot Motion Planning. Kluwer Academic Publishers (1991)
18. LaValle, S.M.: Planning Algorithms. Cambridge University Press (2006)
19. Leroy, S., Laumond, J.P., Simeon, T.: Multiple path coordination for mobile robots: A geometric algorithm. In: Proccedings of the International Joint Conference on Artificial Intelligence (IJCAI, pp. 1118–1123 (1999)
20. Likhachev, M., Ferguson, D.I., Gordon, G.J., Stentz, A., Thrun, S.: Anytime dynamic A*: An anytime, replanning algorithm. In: Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS), pp. 262–271 (2005)
21. Maffei, R.Q., Botelho, S.S.C., Silveira, L., Drews Jr., P., Duarte Filho, N.L., Bicho, A.L., Almeida, F.R., Longaray, M.M.: Space D*: Um algoritmo para path planning multi-robs. In: Proceedings of the VIII ENIA/CSBC, pp. 607–618 (2011)
22. López de Màntaras, R., Amat, J., Esteva, F., López, M., Sierra, C.: Generation of unknown environment maps by cooperative low-cost robots. In: Proceedings of the First International Conference on Autonomous Agents, pp. 164–169 (1997)
23. Okada, T., Beuran, R., Nakata, J., Tan, Y., Shinoda, Y.: Collaborative motion planning of autonomous robots. In: Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing, pp. 328–335 (2007)
24. Pallottino, L., Scordio, V.G., Frazzoli, E., Bicchi, A.: Decentralized cooperative policy for conflict resolution in multi-vehicle systems. IEEE Transactions on Robotics **23**, 1170–1183 (2007)
25. Parsons, D., Canny, J.: A motion planner for multiple mobile robots. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp. 8–13 (1990)
26. Pereira, G.A.S., Das, A.K., Kumar, V., Campos, M.F.M.: Decentralized motion planning for multiple robots subject to sensingand communication constraints. In: Proceedings of the Second MultiRobot Systems Workshop, pp. 267–278 (2003)
27. Rekleitis, I.M., Dudek, G., Milios, E.E.: Multi-robot exploration of an unknown environment, efficiently reducing the odometry error. In: Proceedings of the Fifteenth

International Joint Conference on ArtificalIntelligence, pp. 1340–1345 (1997)
28. Runions, A., Fuhrer, M., Lane, B., Rolland-Lagan, P.F.A.G., Prusinkiewicz, P.: Modeling and visualization of leaf venation patterns. ACM Transactions on Graphics (TOG) **24**, 702–711 (2005)
29. Stentz, A.: The focussed D* algorithm for real-time replanning. In: Proceedings of the International Joint Conference on Artificial Intelligence (1995)
30. Wagner, D., Schmalstieg, D.: ARToolKitPlus for pose tracking on mobile devices. In: Proceedings of the 12th Computer Vision Winter Workshop (CVWW), pp. 139–146 (2007)
31. Wurm, K.M., Stachniss, C., Burgard, W.: Coordinated multi-robot exploration using a segmentation of the environment. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots andSystems, pp. 1160–1165 (2008)
32. Yamauchi, B.: Frontier-based exploration using multiple robots. In: Proceedings of the Second International Conference on AutonomousAgents, pp. 47–53 (1998)