An Efficient Path Planner for Large Mobile Platforms in Cluttered Environments

Tarek Taha, Jaime Valls Miró and Dikai Liu
ARC Centre of Excellence for Autonomous Systems
Mechatronics and Intelligent Systems Group
University of Technology Sydney
NSW2007, Australia
{t.taha, j.vallsmiro, d.liu}@cas.edu.au

Abstract— **This paper presents a one step smooth and efficient path planning algorithm for navigating a large robotic platform in known cluttered environments. The proposed strategy, based on the generation of a novel search space, relies on non-uniform density sampling of the free areas to direct the computational resources to troubled and difficult regions, such as narrow passages, leaving the larger open spaces sparsely populated. A smoothing penalty is also associated to the nodes to encourage the generation of gentle paths along the middle of the empty spaces. Collision detection is carried out off-line during the creation of the configuration space to speed up the actual search for the path, which is done on-line. Results prove that the proposed approach considerably reduces the search space in a meaningful and practical manner, improving the computational cost of generating a path optimised for fine and smooth motion.**

I. Introduction

The problem of computing a collision-free path for a moving object among obstacles is well known in the field of robotics, and has been an active research topic for decades [4]. The majority of the proposed algorithms transform the problem into a pure geometric path planning problem by defining the search in what is known as configuration space, or C-space, an approach originally introduced by Lozano-Perez [2], [5]. Here, a robot with k degrees of freedom can be described by k values, which can in turn be considered as a single point in a k-dimensional C-space of the robot. This configuration is considered free if two parts touch and blocked when two parts overlap. For mobile robots operating in flat ground, C-space is usually defined as a set of all possible configurations encoding the position and orientation of the vehicle. A collision free feasible path is that connecting the start and goal point configurations. Also, a holonomic characteristic is normally assumed, which holds for the case of differential-drive robots like a wheelchair.

The exact construction of the C-space is however a computationally expensive solution to the path planning problem. The need to move away from complete path planning algorithms inspired the development of sampling-based techniques. Hence, the majority of techniques make further assumptions and construct approximate representation of the C-space using sampling-based techniques. These techniques provided a faster practical solution by sacrificing completeness, in which a set of sampling points are used to represent the C-space that is used in constructing solutions. Traditionally, sampling-based algorithms are based on uniform sampling which considers the whole environment as uniformly complex and thus the overall sampling density will be equivalent to the density needed by the most complex region. The result is that every region in the C-space has the same computational complexity, reaching its worst case when narrow passage areas exist in the environment [1]. Furthermore, paths produced by randomised planners usually contain non-smooth segments because of this randomness and the absence of optimisation criteria.

For the problem of navigating large robots in narrow and cluttered environments, such as as ”intelligent” wheelchairs in the average home surroundings, conventional path planning algorithms based on free C-space construction also tend to fail: in order to be able to consider the robot as a k-dimensional point, they generally expand the obstacles in an over-simplistic manner by the length of the larger robot dimension, which very often will prevent reaching a solution even when it exists [5].

In this paper we propose a hybrid path planning algorithm inspired by the C-space approach, where we avoid the computational complexity of generating a denser search area by employing a non-uniform sampling density: this is increased in complex areas, leaving simple areas with lower resolution density, hence directing computational resources towards the complex areas, also know as narrow passages. A reduction of the information embedded in the C-space, and a smoothing cost function are also introduced to generate smoother paths in an efficient manner. The algorithm takes further advantage of techniques like the bridge test [3] and an optimised obstacle expansion method to further reduce the number of samples and the points to be check for obstacle collision. A modified A* search is then implemented to find suitable paths on this space.

The remainder of this paper is organised as follows: latest proposals to the path planning problem and where our approach represents an improvement for the problem at hand is analysed in-depth in Section II. The proposed methodology for the creation of the search space is presented in Section III, with Section III-A.5 explaining the non-uniform random discretisation. Section III-B summarises the customisations to the A* graph search technique to take advantage of the search framework proposed here. Detailed experimental setup and results with simulation and a real wheelchair robot are provided in Sections IV and IV-A respectively. Finally, Section V summarises the contribution of this paper.

II. Background

In general terms, the sampling algorithms developed to construct an approximate representation of the free space currently available can be divided into two: single-query and multiple-query approaches. Multiple-query approaches starts with a pre-processing step that usually takes a large amount of time but makes solving path planning problems in the same environment faster. Probabilistic Roadmaps (PRMs) [6] is an example of a multi-query approach that initially used uniform sampling in constructing the path. This method was problematic because the entire C-space will be sampled with a density required by the most complex area of the environment, such as a narrow passage area. Nowadays, PRMs are moving into a non-uniform methods for sampling such as the Gaussian sampling method [10], and the bridge test to insure that most of the configurations in C-space are actually close to obstacles or inside a narrow passage, thus reducing the unnecessary samples and decreasing the computational time. Single-query methods were developed to avoid the large pre-computational time that the multi-query methods take, and they have been proved to be efficient [15], [16]. Randomly-exploring Random Trees (RRTs) [12], [18] are mainly based on single-query methods. They have gained popularity from their good performances, which has lead to a number of extensions specifically targeting the solution to complicated geometrical problems [17], such as the deterministic resolution-complete alternatives that have been proposed to replace the random sampling methods in [19].

In many cases, an optimal and not just a feasible path is required. As a result of randomness, the paths generated by the execution of the above planners are very often sub-optimal and non-smooth. A two-phase approach was proposed in [8] to optimise paths generated in the special case where the first-phase path planned is made up of straight line segments connected by way-points. Another two-phase planning algorithm based on RRT was developed in [9]. This algorithm can compute low cost paths given a desired cost function by a numerical gradient descent algorithm that minimises the Hamiltonian of the entire path.

The approach proposed here is based on a simple multi-query one-phase planner that addresses the optimality and smoothness weaknesses of probabilistic path planning algorithms, a requirement for the general case of fine motion platforms, such as wheelchairs. The planner uses an a-priory map of the environment to calculate an offline, minimal free search space, where and additional smoothness cost function is used to address the issues associated with smoothly navigating a large robot in an environment with narrow passages and obstacles.

III. Path planning algorithm

The proposed algorithm starts by generating the search space to containing information about the node position, the connections to neighbouring nodes, a path smoothing penalty which will be used later to fine-tune the path during the on-line planning, and a collision detection method. The on-line path planning step consists of a path search using the A* algorithm with a modified cost function to favour smooth fine motions. The result is an optimal and smooth path that can be quickly generated in one step.

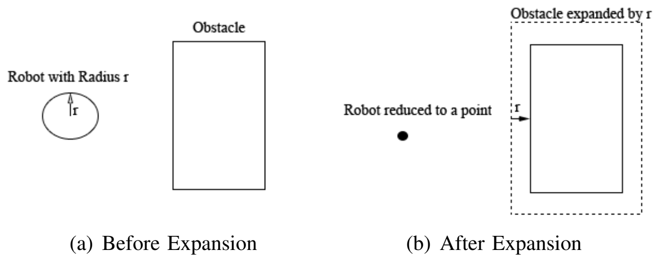

Fig. 1. Largest robot dimension obstacle expansion method

A. Generating the search space

The pre-processing step aims at minimising the on-line computation by pre-generating a search space to contain all the information that will be used during the on-line path planning, while at the same time avoid generating an unnecessarily complete and complex space. The steps used during the search space creation can be defined as follows:

Algorithm 1 C-space generation

```
Input: map, robot dimensions
1. Expand Obstacles.
2. Generate Regular Grid with low resolution.
3. Apply bridge test to add dense narrow passages.
4. Penalise nodes by adding a smoothing cost.
5. Connect nodes to form search space discretisation.
6. Eliminate those that cause collision.
Output: free search space.
```

1) Obstacle expansion: In this step obstacles are enlarged with a radius R to simplify the on-line collision detection by reducing the number of points on the robots to be checked for collision. It also reduces the number of nodes in the search space thus increasing the on-line path planning performance. The traditional approach [14] is based on expanding the obstacles by a radius r equivalent to the robots largest dimension, hence planning as if the robot could navigate as a point in the environment, as depicted in Figure 1. This over-simplification, however, is not suitable for the case of large robots in constrained spaces, as expanding the obstacles along narrow passages will effectively block the passage, as shown in Figure 2.

A more suitable solution is proposed by finding the largest possible expansion radius R that allows the robot to pass through the narrowest path and then divide the area of the robot into circles of that radius, as depicted in Figure 3. The centre points of those circles will then be used to check for

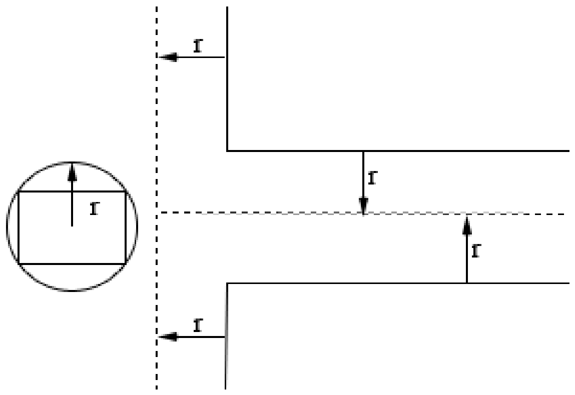

Fig. 2. Narrow passage blocked as a result of largest robot dimension obstacle expansion

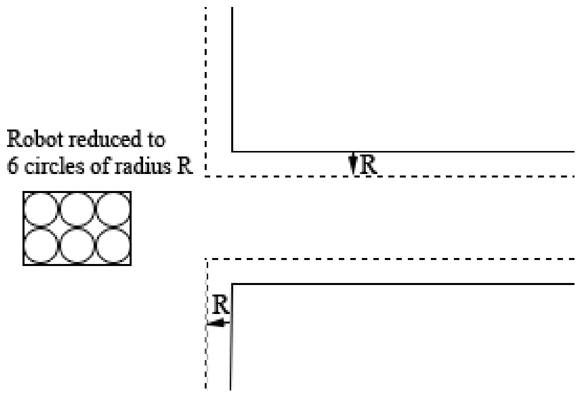

Fig. 3. The area of the robot is covered by circles of radius R, the centres of these circles will be the points to be checked for collision

obstacle collision. The expansion radius R is determined based on the a-priory knowledge of the environment: suppose that the narrowest passages is of width l and the largest robot dimension is r then the largest expansion that allows the robot to pass through can be determined by:

$$R = \begin{cases} \frac{l-\varepsilon}{2} & \text{if } l < 2r \\ r & \text{otherwise} \end{cases} \tag{1}$$

where ε is a minimal safety distance to make sure the platform does not get uncomfortably close to the obstacles.

2) Regular grid discretisation: The C-space is then populated with nodes using a low resolution regular grid. This will help in maintaining the connectivity of the graph by defining a minimum discretisation for the open spaces. The discretisation density is adjusted to suit the environment, selecting as sparse a grid as possible. Up to this stage the nodes hold only position information.

3) Bridge test: Narrow passages in free space are small regions critical in preserving the connectivity of a path during the path planning process. Any attempt to sample the narrow passages using a uniform distribution based on volume will fail precisely because of their small volumes. The bridge test [3] was introduced to boost the sampling density inside narrow passages using only a simple test of the local geometry. Narrow passage can be usually defined as a space where the motion of the robot is restricted in at least on direction and any small changes in the robots configuration in that direction may result in a collision with obstacles. In these passages, robot motion is limited to those directions perpendicular to the restricted ones. A short line segment of length d can sample randomly through a point m in the free space such that the end points of the line segment lie in obstacles. This line segment is what we call a bridge because it acts like a bridge across the narrow passage with its endpoints in an occupied location and the point m in a free space. If we are able to build a bridge through point m, then the bridge test is successful at this point and point m is added to the search space.

Algorithm 2 Bridge Test

```
1. repeat
2. Pick a point $p$ from the regular-grid map
3. **If** $p$ is in an occupied location **then**
4.     Pick a point $p'$ that is $d$ distance away from $p$
5.     **If** $p'$ is in an occupied location **then**
6.         Let $m$ be the midpoint of $pp'$ line segment
7.         **If** $m$ is in a free location **then**
8.             Insert $m$ into the search space as a new node
```

Building short bridges is easier in narrow passages than in free space and by favouring short bridges we increase the chance of getting point in the narrow passages. The off-line test increases the density of free-space sample points to our search space where it matters most, in the narrow passages, instead of the whole region. Figure 5 (further described later in Section IV) shows the result of the bridge test on a map with narrow passages.

4) Clearance (smoothness) penalty: Costs are added to the nodes in C-space to indicate how far they are from an obstacle. The cost C_p is a normalized cost that is inversely proportional to this distance d, so that the closer the point is from an obstacle the higher its cost, according to:

$$C_p = \frac{D-d}{D} \tag{2}$$

where D indicates the clearance distance beyond which the node will be assigned a zero cost. This cost will be used during the on-line path planning process to plan smoother paths in one step, and no further smoothing step will be necessary after the path is generated. The end results are paths which tend more towards the middle of empty spaces and are within a safe distance from the obstacles.

5) Node connections: In order to find a path among the resulting nodes, these need to be connected together. This is done by establishing a link between each node and its neighbouring nodes a certain distance away. The more neighbours a node is linked to, the more discrete poses (position and orientation) with be available during the search for a viable path. However, there should be a compromise between the connecting distance, and the computational complexity: shorter distances between the nodes will result in fewer angular and position discretisations (fewer neighbours) and less impact on computation, but might decrease the possibility of finding a path. On the other hand, longer distances will produce a larger number of connections, hence increasing the chances of finding a solution at the expense of complexity. A reasonable compromise is to connect nodes with a distance equivalent to the distance used to sample the uniform regular grid. This ensures continuity in node connections and at the same time results in a faster search.

6) Collision Detection: Finally, those connections that cause a collision of the platform with the obstacles are eliminated. The connections between nodes determines the possible orientations of the robot should it follow that path. The center of the circles that describe the area of the robot along that path can be rotated and translated accordingly. Hence we can then determine if any of them falls into an occupied area or not, removing those conecting nodes that will cause a collision. The result will be a collision free search space of connected nodes in which to efficiently carry out the on-line search for a path.

B. On-line Path Planning

The A* path planning algorithm [13] is a well known technique, well regarded for its accuracy and calculation speed in searching for an optimal solution. A* works by exploring nodes based a cost function which is the sum of g(n), the cost from the start node to node n, and the estimated cost from node n to the goal h(n). It uses an heuristic search to estimates the cost to the goal node and minimises the cost of the path so far. A* is optimal if the estimated cost to the goal is always underestimated. Since the shortest distance between two points is a straight line, euclidean distance serves as an excellent estimated cost to goal, making A* well suited for fast computations. In the algorithm proposed here, the cost function $J(d)$ combines the sum of the partial path distances Δd, the sum of all the distances travelled as a result of changing orientation $\Delta\theta\, x$ where x is the length of the axis of the rear wheels, the sum of the clearance penalties previously computed offline - which is directly proportional to the distance Δd - and the number of reversals (backward motion) in the path n_{rev}. The cost function, defined as follows:

$$J(d) = \sum \Delta d + \sum \Delta\theta\, x + \sum \Delta d\, C_p + \sum n_{rev} \quad (3)$$

encourages the robot to avoid whenever possible turns and reversing actions, while at the same time directing it towards the middle of free space. The result is a smooth and secure path - in the context of the obstacles around the platform - efficiently generated in a single step.

IV. Experimental Results

In order to compare the efficiency of the proposed algorithm, we first compare the performance of a traditional uniform regular sampling C-space with our random non-uniform sampling approach for a simple navigation problem.

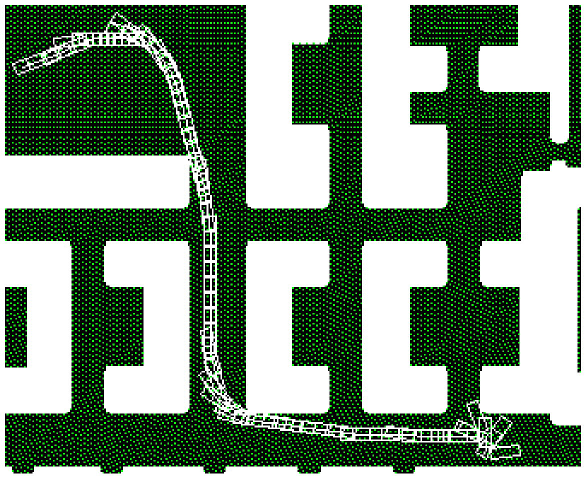

Fig. 4. Uniform sampling: the whole environment is equally mapped with a constant density of search nodes

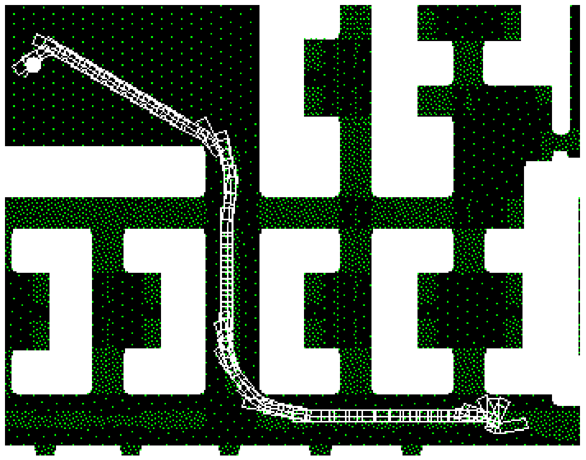

Fig. 5. Non uniform sampling: density is increased around tight places, leaving open spaces with a lower number of search nodes

The simulations were carried out in a PC with a 1.8 GHz Pentium IV processor and 512 Mb RAM. A map of size $45x20m$ was discretised uniformly by $0.1m$ to generate the search space, a detailed section of which is shown in Figure 4, where white spaces represent the obstacles in the map after having bee expandedn. This is the same density employed to discretised the tight passages with the non-uniform sample method proposed here, where a bridge test with $2m$ length was exercised. The resulting search map is that depicted in Figure 5, where a $0.2m$ uniform sampling density was employed for the open spaces. In both cases the obstacle expansion was set to $0.2m$, and the starting and goal configurations were the same. Search technique was also the same in both cases, so that the final paths produced were expectedly similar, as seen by the sequence of rectangles which represent the configurations of the wheelchair at each step on the planned paths based on the wheelchair footprint. However, the comparison results in table I clearly show the computational advantages of the random sampling technique proposed here.

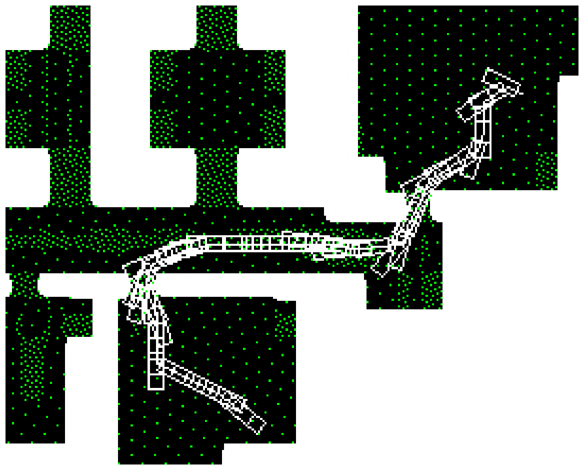

Fig. 6. A case where the wheelchair robot has to go through two narrow doors

TABLE I

COMPARISON OF UNIFORM AND NON-UNIFORM C-SPACE GENERATION

	Uniform Sampling	Random Sampling
no. nodes	32364	15634
no. connections	691155	129479
time	1.8589 sec	0.3228 sec

A more challenging path planning problems where the mobile robot had to manouvre to pass through two tight and narrow passages is depicted in Figure 6. It can be seen how the addition of search nodes where it really matters not only makes finding a collision-free path feasible, but the additional penalties also tend to direct the robot as far as possible from obstacles along a smooth path.

A. *Testing on a hardware platform*

The real-time feasibility and smoothness of the approach was also tested in a real mobile platform, a commercial electric-powered wheelchair. The platform has two differentially driven wheels at the rear, and two passive casters at the front, and can travel at speeds of up to 15km/h. The wheelchair, depicted in Figure 7, was instrumented with a computer (attached behind the back rest), wheel encoders and a laser range finder used for localisation. The functional architecture of the system is described by the block diagram of Figure 8.

The computing platform comprises of a 1 GHz Pentium III processor with 256Mb RAM that communicates with the two optical wheel encoders through the serial port. A laser rangefinder is located on the foot rest at the front of the wheelchair and also communicates via a serial link with the computer. The actual control of the wheelchair motion takes place through an interface DAC box that is attached to the joystick and simulates the standard command signals that control the normal funcioning of the wheelchair as if a user was controlling it, such as activating the motors, increasing/decreasing the gears or sounding the horn amongst others. The wheelchair measures 1.2x0.7m, by all accounts a

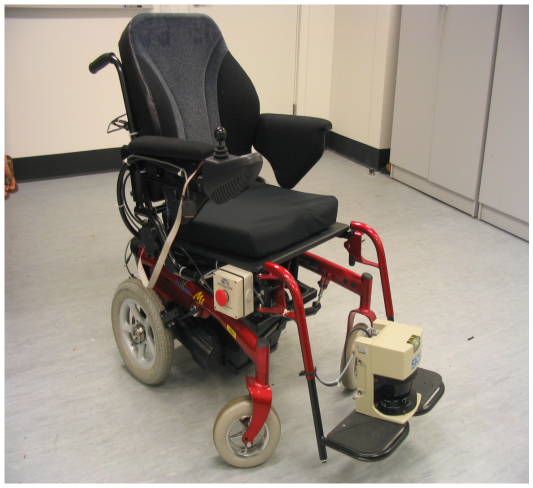

Fig. 7. Instrumented autonomous wheelchair platform

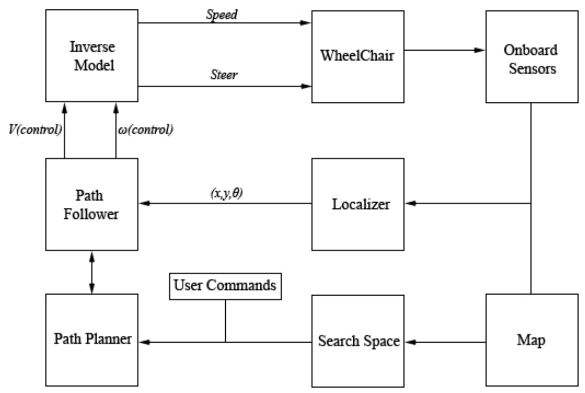

Fig. 8. Mobile platform navigational architecture

large robot when driving around a typical office environment with narrow passages, long corridors and cluttered static obstacles. Tests were fully autonomous in that the wheelchair was commanded to plan a path in the given map from a start configuration to a goal configuration. A simple linear controller based on displacement and orientation error was implemented to traverse the path.

Figure 9 shows the result of navigating a similar path to that depicted in Figure 5. Velocity of the wheelchair was constant during the experiment at 0.2 m/sec. Localising the robot was done through the Adaptive Monte Carlo Localization (AMCL) algorithm [7] with the aid of the laser range-finder. A true estimation of the location was provided every 2 seconds, and dead-reckoning based on odometry and the kinematic vehicle model was employed in between these updates to establish the location of the vehicle. The goal was reached as shown in Figure 9 shows how close the path was followed, achieving the goal position within 3 cm and 5 degrees linear and angular error. Similar results were obtained for other paths generated

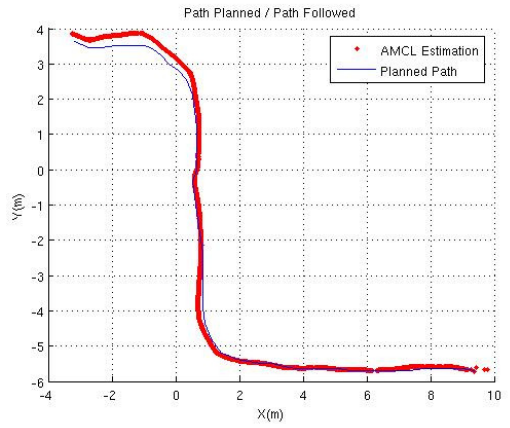

Fig. 9. Result of navigating a path generated by the planner, where the thick line represents the feedback position as estimated by the localizer, and the thin line represents the actual planned path

in this environment.

V. Conclusion and further work

This paper has presented a new approach for generating optimized smooth paths afor large platforms in constrained spaces using a novel C-space creation method. A non-uniform sampling technique has been proposed to efficiently target the narrow passage problem, of particular relevance for such large mobile platforms in cluttered environments. Results from simulation and a real-time implementation in an automated wheelchair have shown that the path planner was able to plan one-stage smooth feasible paths, quickly and efficiently due to the simplicity of the search space method prposed.

While these preliminary results have shown the feasibilty of the motion planner to generate and navigate suitable paths in what represents a challenging, if static, environment for a large mobile platform, the next step is the introduction of dynamic obstacles, which is by no means trivial. Further, the current cost function gives preference to on-the-spot rotations, circumventing non-holonomic constraints which could also be taken into account. Also, efforts are being directed to apply the advances presented here to the fine motion planning and control of a user-driven wheelchair, effectively allowing the user think he/she is a better driver than he/she really is.

Acknowledgment

This work is supported by the Australian Research Council (ARC) through its Centre of Excellence programme, and by the New South Wales State Government. The ARC Centre of Excellence for Autonomous Systems (CAS) is a partnership between the University of Technology Sydney, the University of Sydney and the University of New South Wales.

References

[1] D. Hsu, L. E. Kavaki, J. C. Latombe, R. Motwani and S. Sorkin. On finding narrow passages with probabilistic roadmap planners. In *Proceedings of the Workshop on the Algorithm Foundations of Robotics*, pages 141-154. A K Peters, 1998.

[2] R. A. Brooks and T. Lozano-Perez. A Subdivision algorithm in configuration space for findpath with rotation, IEEE Transactions on Systems, *Man and Cybernetics*, Vol. SMC-15, No. 2, March/April 1985, pp.224-233. Also in *Proc. Eighth Int. Joint Conf. on Artificial Intelligence*, Karlsruhe, W. Germany, August 1983, pp.799-806. Also MIT AI Memo 684, February 1983.

[3] D. Hsu, T. Jiang, J. Reif, and Z. Sun. The bridge test for sampling narrow passages with probabilistic roadmap planners. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2003, pp. 4420-4426.

[4] J. C. Latombe. Motion planning : A journey of robots, molecules, digital actors, and other artifacts. *International Journal of Robotics Research*, vol. 18, no. 11, pp. 1119-1128, 1999.

[5] T. Lozano-Perez. Spatial planning: a configuration space approach. *IEEE Transactions on Computers*, 1983, vol. C-32, pp. 108-120, IEEE Press.

[6] L. E. Kavraki, P. Svestka, J. C Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566-580, 1996.

[7] D. Fox, W. Burgard, F. Dellaert and S. Thrun. Monte carlo localization: efficient position estimation for mobile robots. In *AAAI/ IAAI*, pp. 343-349, 1999 .

[8] J. M. Philips, N. Bedrossian and L. E. Kavraki. Guided expansive spaces trees: a search strategy for motion and cost-constrained state spaces . *Proc. IEEE Int. Conf. Robot. & Autom.* , pp. 3968-3973, 2004.

[9] S. G. Vougiouksa. Optimization of robot paths computed by randomised planners.*Proc. of Int. Conf. on Robot. & Autom.*,pp. 2160-2165, 2005.

[10] V. Boor, M. H. Overmars and A. F. van Der Stappen. The gaussian sampling strategy for probabilistic roadmap planners. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*,pp. 1018-1023, 1999.

[11] H. Chang and T. Y. Li. Assembly maintainability study with motion planning. *IEEE international Conference on Robotics and Automation*, 1995, pp. 1012-1019.

[12] S. M. LaValle. Rapidly-exploring random trees: a new tool for path planning. TR 98-11, *Computer Science Dept., Iowa State University*, Oct. 1998.

[13] J. Barraquand, L. E. Kavraki , J. C. Latombe, T. Y. Li, R. Motwani and P. A. Raghavan. Random sampling scheme for path planning. *International Journal of Robotics Research*, vol.16 no. 6, pp. 759-774, 1997.

[14] P. E. Hart, N. J. Nilsson and B. Raphael. A formal basis for the heuristic determination of minimum cost paths in graphs. *IEEE Transactions on Systems Science and Cybernetics*, vol. 4 no. 2, pp. 100107, 1968.

[15] D. Hsu, J. C. Latombe and R. Motwani. Path planning in expansive configuration spaces. *Int. J. Comput. Geom. & Appl.*, vol. 4, pp. 495-512, 1999.

[16] G. Sanchez and J. C. Latombe. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In *Int. Symp. Robotics Research 2001*.

[17] P. Choudhury and K. Lynch. Trajectory planning for second-order under actuated mechanical systems in presence of obstacles. In *Proceedings of the Workshop on Algorithmic Foundation of Robotics*, 2002.

[18] P. Cheng and S.M. La Valle . Resolution complete rapidly exploring random trees . In *Proc. IEEE Intl. Conf. on Robotics and Automation*, pages 267-272, 2002.

[19] L. E. Kavraki and P. Ostrowski. Motion planning of aerial robot using rapidly-exploring random trees with dynamic constraints. In *IEEE Int. Conf. Robot. & Autom.*, 2003.