SICE Annual Conference 2014
September 9-12, 2014, Hokkaido University, Sapporo, Japan

RRT-A* Motion Planning Algorithm for Non-holonomic Mobile Robot

Jiadong Li[1,2], Shirong Liu[1]*, Botao Zhang[1], Xiaodan Zhao[1]

[1]School of Automation, Hangzhou Dianzi University, Hangzhou, Zhejiang 310018, China
(Tel : +86-0571-86878535; E-mail: liushirong@hdu.edu.cn)
[2]Institute of Automation, East China University of Science & Technology, Shanghai 200237,China
(Tel : +86-0571-86878535; E-mail: lijiadong106305@126.com)

Abstract: The RRT algorithm can deal with the motion planning problems in consideration of non-holonomic differential constraints, but it does not taken into consideration the optimal path problem in planning process. Random selection of nodes leads to every planning cost relatively different because of only use of metric function in new node selection. In this paper, an improved heuristic RRT-A* algorithm is proposed for robot motion planning with non-holonomic constraints. In this algorithm, the cost function of A-Star(A*) is introduced into the RRT algorithm to optimize the performance. Meanwhile, several metric functions are used as the heuristic information functions respectively to measure the performance of different metric function. The simulation results shown that the Manhattan heuristic information function based RRT-A* planning algorithm is better than the other improved RRT algorithms in optimization path and computational cost.

Keywords: Mobile robot; non-holonomic constraint; motion planning; Rapidly-exploring Random Trees, A-Star

1. INTRODUCTION

Path planning plays a paramount role in autonomous robot navigation. It ensures to find a trajectory from the initial point to the destination, subject to rules of motion and any other constraints, such as collision avoidance, balance and joint limits[1]. There are several types of problem depending on the types of constraints.

In the early stage, motion planning is a simple geometric path planning problem considering the geometric constraint, for example, the obstacle constraint, aims to find out a geometric curve from the starting point to the target. It is a simple problem of kinematic discipline, and is similar to the "Piano Mover's Problem"[2]. For the low complexity of the planning configuration and regardless of the concept of time, this type of motion planning problem is easy to implement, just needs to find out a sequence of geometric position mobilizing and reversing, regardless of velocity and acceleration constraints.

Since the motion planning problem considering non-holonomic differential constraints was taken into the motion planning problem for robot[3], it has received a considerable amount of attention, especially in the last decade. The non-holonomic differential constraint is a kind of local constraint, which limits the moving speed of wheeled mobile robot in mobile robots, is different from the obstacle constraint caused by global constraint. Now the even more complex motion planning problem, which considers the constraints called kino-dynamic differential constraints[4], comes into our view. It is used to deal with the motion planning problem like common bicycle motion planning. Except for the non-holonomic differential constraint, we also have to consider the balance problem of robot body. This leads to the motion planning problem becomes complicated.

Traditional path planning methods ignore these types of constraints in process of planning, and then deal with it when following the tracks. It requires designing a control input function to ensure the robot following the path accurately. But this is just a feasible way, meanwhile increases the complexity of algorithm. The better way to deal with differential constraint is considering it in planning process.

Random sampling based planning algorithm like RRT can solve motion planning problem, and take the differential constraint into consideration [5]. The performance of RRT single-query makes it be popular in high dimension and complex environment. Lots of studies have been emerged since RRT algorithm was adopted, there are various modifications of the proposed algorithm RRT, for example, Goal-biasing-RRT[6]. Even though these algorithms are not complete, they provide probabilistic completeness to guarantee planning successfully as much as possible. When there is a feasible path, as the number of sampling nodes tends to be infinite, the probability of failure to find this path will decay to zero exponentially.

However, the node random selection leads to different planning cost. In this paper, an improved heuristic RRT-A* algorithm is proposed for robot motion planning with non-holonomic constraints. the A* [7] cost function is introduced into the RRT algorithm to optimize the path performance. Meanwhile, several metric functions are used as the heuristic information functions respectively to measure the performances of RRT-A* algorithms with different metric functions.

2. MOTION PLANNING FOR NON-HOLONOMIC ROBOT

2.1 Non-holonomic differential constraints

The non-holonomic system can be formulated as [8]

$$\begin{aligned} \frac{dq}{dt} &= F(q, u, t) \\ C(q, \dot{q}, t) &= 0 \end{aligned}, \qquad (1)$$

Where $C(q, \dot{q}, t) = 0$ is the constraint equation, $q \in R^n$ denotes the state variable, $u \in R^n$ is the control variable, and t is the time variable. The above non-holonomic system does not have a function that can make $dG(q,t)/dt = C(q,dq/dt,t)$ be tenable. It means that the system cannot be expressed completely without the usage of derivatives.

The wheeled system is one of the most popular mobile platforms in mobile robots. This kind of robots has to roll along a special direction and cannot skid in the process of moving, which means the speed vector of wheel is restrained. The constraint cannot reduce the degree of freedom to the system configuration. Therefore, the number of action variable is less than the dimension of system. The non-holonomic differential constraint makes the motion planning problem complicated. A path planned by the planner is unable to be tracked by non-holonomic wheeled vehicles if the non-holonomic differential constraint is not taken into consideration.

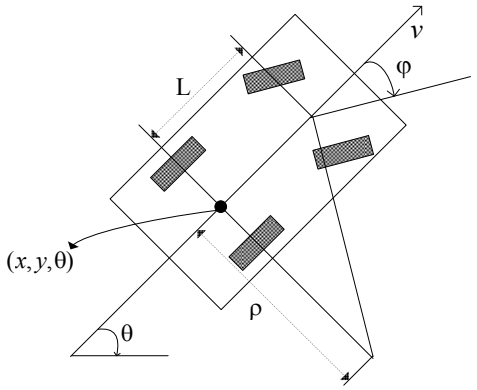


Fig. 1 A model of car-liked robot

Figure 1 is the simple model of a car-liked robot, in which the non-holonomic differential constraint is considered[9]. The robot motion is assumed as a rigid body in the configuration space $X \in R^2 \times S^1$ .The configuration can be expressed as $\vec{q} = f(x,y,\theta)$, where (x,y) is the coordinate of the robot, θ denotes the angle between the coordinate system and the moving direction. The speed v and the steering angle φ are both inputs. When Δt decays to zeros, the non-holonomic differential constraint of the system can be expressed as $dy/dx = \tan\theta$. Take the two-dimensional action variable as $\vec{u} = (u_v, u_\varphi)$, where $u_v = v, u_\varphi = \varphi$. The robot's state equation can be expressed as follows

$$\begin{cases} \dot{x} = u_v \cos\theta \\ \dot{y} = u_v \sin\theta \\ \dot{\theta} = u_v \tan u_\varphi / L \end{cases} . \quad (2)$$

2.2 Non-holonomic motion planning

The ultimate goal of non-holonomic motion planning is not only generating a trajectory, but also the input control sequence for trajectory tracking. The main difference between non-holonomic and holonomic motion planning is the state transition of robot. For the mobile robot whose configuration is $q=(x,y,\theta)$, the state transformation is simply linear replacement when just considering holonomic and geometric constraint. The linear replacement can be expressed as:

$$q_{k+1} = F(q_k, u_k), \quad (3)$$

Considering the non-holonomic system shown in figure 1, we should have an integral instead of a linear replacement. The Euler's integral is formulated as

$$q_{new} \approx q + \int_0^{\delta t} f(q,u)dt . \quad (4)$$

Where $f(\cdot)$ is the equation of state just like the formula (2).

3. AN IMPROVED RRT MOTION PLANNING ALGORITHM

3.1 The RRT algorithm

```
RRT( )
1  Tree_init(q_init);
2  while !flag do
3      q_rand ← SamplePoint();
4      q_nearest ← Nearest_node(T, q_rand);
5      q_new, u_best ← Control(q_nearest, q_rand);
6      if collisionfree
7          tree ← tree ∪ {q_new};
8      if ||q_new − q_goal|| ≤ d
9          return flag=true;
10     else return flag=false;
11 function Control(q_nearest, q_rand)
12     d_min ← ||q_nearest, q_rand||, u_best ← ∅
13     for u ∈ U do
14         q_new ← integrate(q_nearest, u);
15         if collision(q_nearest, u, q_new)
16             next u;
17             d ← ||q_new, q_rand||;
18             if d < d_min then
19                 d_min, u_best ← d, u
20     return q_new, u_best;
```

Fig. 2 The Pseudo code of RRT Algorithm

RRT algorithm is a motion planning algorithm based on random sampling. It can explore the whole space rapidly by expanding branches from the neighboring node to the random target. The property of single-query makes RRT can be applied to solve the motion planning problems in highly-constraints environment and the problems with high dimension perfectly. The pseudo code of RRT algorithm is as shown in Fig. 2.

3.2 Metric Function Problem

When choosing the nearest neighbor q_{nearest} as the following best input variable, the original RRT adopts the principle of nearest neighbor accomplished by the metric function. And there are several metric functions used in RRT algorithm as follows, only in the space of two dimensions.

1) The Euclidean distance metric function. The Euclidean distance between two points can be expressed as:

$$d_{\text{Euclid}}(\mathrm{i},\mathrm{j}) = \sqrt{(x_{i1}-x_{\mathrm{j}1})^2+(x_{i2}-x_{\mathrm{j}2})^2} \quad (6)$$

2) The Manhattan distance metric function. The Manhattan distance between two points can be expressed as:

$$d_{\text{Manhattan}}(\mathrm{i},\mathrm{j}) = |x_{i1}-x_{\mathrm{j}1}|+|x_{i2}-x_{\mathrm{j}2}| \quad (7)$$

3) The Diagonal distance metric function. The diagonal distance is the compromise scheme of Manhattan distance which can be expressed as

$$d_{\text{Diagonal}}(\mathrm{i},\mathrm{j}) = \max(|x_{i1}-x_{\mathrm{j}1}|,|x_{i2}-x_{\mathrm{j}2}|) + (\sqrt{2}\text{-}1)\min(|x_{i1}-x_{\mathrm{j}1}|,|x_{i2}-x_{\mathrm{j}2}|) \quad (8)$$

However, the metric function only guide the extension of tree far enough[10]. For the holonomic planning problem, the distance between dots can be solved by the metric functions. But for the non-holonomic system considering the non-holonomic differential constraint, the metric function ignores the global constraint information in the environment. The efficiency can be reduced by the misleading information from metric functions. In addition, the property of random sampling making the otherness of cost very large.

For this problem, the benevolent see benevolence and the wise see wisdom, for example the research in the literature [11], the cell decomposition method is used to divide the global environment into collision zone and free zone in advance of planning. In literature [10], a collision information test mechanism was proposed on the basis of the original metric function. In the collision information, the global constraint is taken into account, and the direction of expanding is guided to the direction whose collision probability is lower. Meanwhile, introducing the cost function into RRT is also a great idea. An ideal cost function can perfectly reflect the performance of the optimal path index, such as the length of path, the running time of algorithm and the consumption of energy. For example, in the literature [12], a cost function based on mechanical work is proposed to solve the path planning problem on configuration-space costmaps. However, calculating the optimal cost-to-goal is at least the same difficulty as the trajectory planning problem.

3.3 A* Cost Function based RRT Motion Planning Algorithm

Cost function of A* algorithm is able to guide the path towards the target, meanwhile guarantees the optimality of path. In this paper, the A* cost function is adopted into RRT to optimize the path. A RRT-A* algorithm is employed to process the non-holonomic motion planning problem. The cost function of A* is shown as formula (5),

$$\mathrm{f(q)=g(q)+h(q)}, \quad (5)$$

$\mathrm{f(q)}$ represents the estimated cost of ultimate path, $\mathrm{g(q)}$ is the actual value of the path which has been planned already, $\mathrm{h(q)}$ is the heuristic information function which represents the estimated cost between the current state q to the target. The ultimate effect of path depends on whether or not $\mathrm{h(q)}$ is selected properly.

Heuristic – *RRT*( )

1 Tree_init(q_{init});
2 while !flag do
3 for $i = 1$ *to* K *do*
4 $\mathrm{q}_{rand}(i) \leftarrow$ *Sample*Point();
5 $\mathrm{q}_{nearest}(i) \leftarrow$ *Nearest _ node*($\mathrm{T}, \mathrm{q}_{rand}(i)$);
6 $\mathrm{f}(i) = \mathrm{f}(\mathrm{q}_{nearest}(i)) = g(\mathrm{q}_{nearest}(i)) + h(\mathrm{q}_{nearest}(i))$
7 $n = \arg\min\_\{i\}\, \mathrm{f}(i)$;
8 $\mathrm{q}_{nearest} = \mathrm{q}_{nearest}(n)$
9 $\mathrm{q}_{new}, \mathrm{u}_{best} \leftarrow$ *Control*($\mathrm{q}_{nearest}, \mathrm{q}_{rand}$);
10 *if collisionfree*
11 tree $\leftarrow$ tree $\cup \{\mathrm{q}_{new}\}$;
12 *if* $\|\mathrm{q}_{new} - \mathrm{q}_{goal}\| \le d$
13 *return* flag=true;
14 *else return* flag=false；

Fig. 3 Pseudo code of Heuristic RRT algorithm based on A* cost function

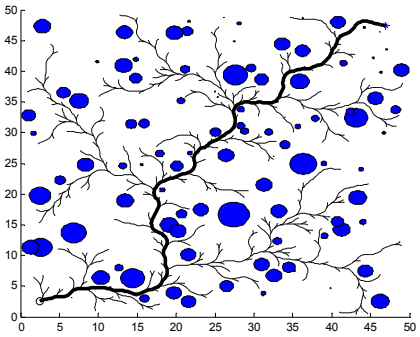
As shown in Fig. 3, the difference between RRT-A* and RRT is the process of the nearest neighbor selection. K-Nearest method is adopted, in which K random targets are collected by random sampling, then the K nearest nodes $\mathrm{q}_{nearest}(i), i = 1,2,...,K$ are searched by function $\mathrm{Nearest\_node}(\cdot)$. The ultimate new direction which will be expanded depends on the A* cost's size of the nearest nodes.

4. SIMULATIONS AND ANALYSIS

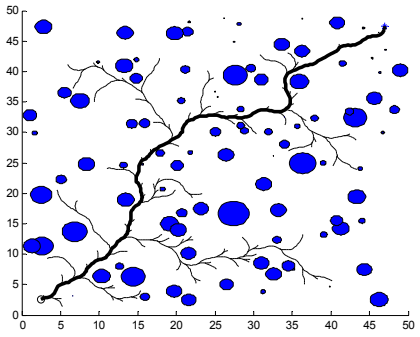
Several algorithms are simulated in the spare (Fig. 4) and dense (Fig. 5) environments with round obstacles, the model constrains of non-holonomic mobile robot shown in Fig.1 are considered in the algorithms. The simulations were carried out in the MATLAB R2010b platform, the computer Intel Core (TM)2(@2.10Ghz、2.10Ghz), 2.00G memory. The model parameters in (2): $\varphi \in (-\pi/6, \pi/6)$, $v = 3\,m/s$, $L = 0.5\,m$, $\Delta t = 0.3\,s$.

4.1 Simulation Results

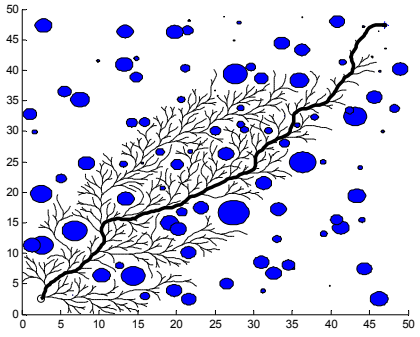
The simulation results for the planning algorithms, such as RRT, Goal-biasing-RRT and RRT-A* with Euclidean function, Diagonal function and Manhattan function, are respectively shown in Fig.4 and Fig.5, and the performance indicators are listed in Table 1, which are the statistical results after 50 runs, and the cost is the length of path.



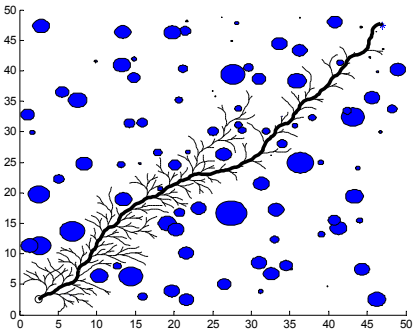
(a) RRT



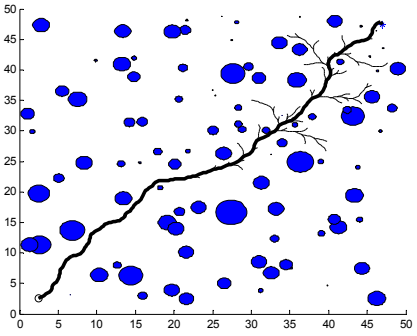
(b) Goal-biasing-RRT



(c) Euclidean heuristic information function based RRT-A*

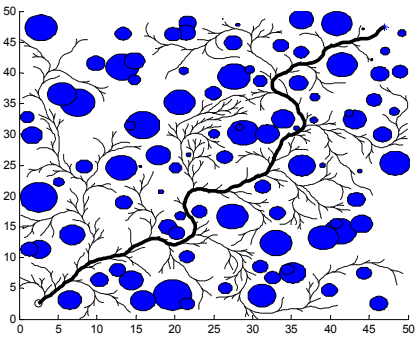


(d) Diagonal heuristic information function based RRT-A*

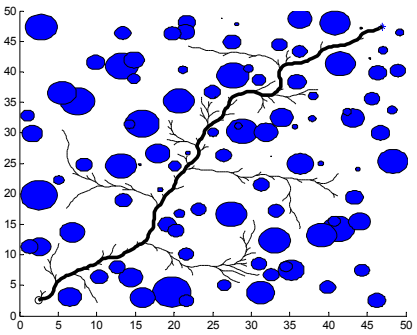


(e) Manhattan heuristic information function based RRT-A*

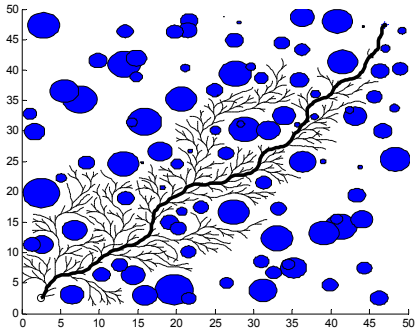
Fig. 4 Planning results in the sparse environment



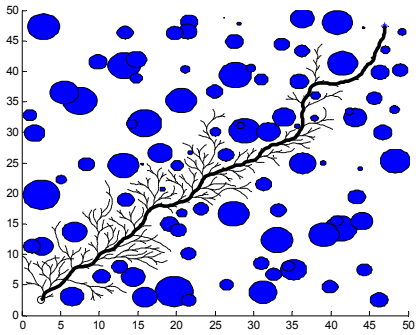
(a) RRT



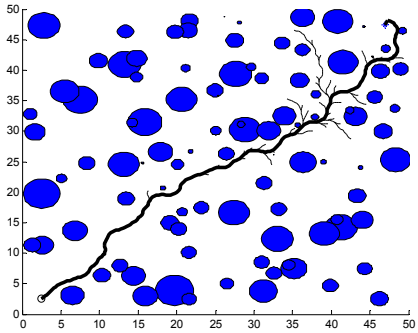
(b) Goal-biasing-RRT



(c) Euclidean heuristic information function based RRT-A*



(d) Diagonal heuristic information function based RRT-A*



(e) Manhattan heuristic information function based RRT-A*

Fig. 5 Planning results in the dense environment

4.2 Analysis

From Fig. 4. Fig. 5 and Table 1, we can obtain the following analysis results.

1) The planning time and nodes of path generated by original RRT are the largest in all the algorithms. The searching routes are very decentralized, distributed throughout the space.

2) The number of nodes is also reduced as a result of goal-biasing affecting, which makes the planning time of goal-biasing-RRT be very short.

3) The cost of path generated by the RRT-A* based on Euclidean distance is minimum in all of them, but the planning time is large, that is because there is need to take a large number of nodes when select Euclidean distance as the heuristic information function.

Table. 1 The simulation data of RRT planner based on several metric functions

Environment	Algorithm	Time(s)	Cost	Nodes
Sparse environment with round obstacles	Euclidean based RRT	22.29	75.65	816
	Euclidean based goal-biasing-RRT	2.86	73.40	337
	Euclidean based RRT-A*	27.14	68.96	838
	Diagonal based RRT-A*	13.56	69.32	658
	Manhattan based RRT-A*	**1.34**	**71.23**	**153**
Dense environment with round obstacles	Euclidean based RRT	28.167	73.62	892
	Euclidean based goal-biasing-RRT	6.745	73.367	429
	Euclidean based RRT-A*	21.73	70.36	781
	Diagonal based RRT-A*	15.85	70.87	668
	Manhattan based RRT-A*	**1.55**	**71.69**	**172**

The number of nodes generated by the RRT-A* based on diagonal distance is decreased, but the cost is a little larger than the former. The path generated by Manhattan distance based RRT-A* looks ideal, the planning time and number of nodes are both small, in spite of its cost which is a little larger.

4) There are a large number of branches on the initiating terminal of random tree which is generated by

RRT-A* based on Euclidean heuristic information function. But RRT-A* based on Manhattan heuristic information function is contrary to the former, its random tree has more branches on the end terminal of random tree. The reason of this difference lies in the different heuristic information distance functions. Distance function can express similarity of signal. For Euclidean distance, two closer nodes have more similar similarity of signal, meanwhile easier to interfere with each other. This situation is contrary to the Manhattan distance. And the diagonal distance is compromise of the former.

Overall, the cost of path generated by RRT-A* is smaller than RRT and goal-biasing-RRT, and the trajectory curves are focus on the area of optimal trajectory. There is a positive correlation between the number of nodes and planning time. The number of nodes depends on the metric function, and then influences the efficiency of planner. The performance of the Manhattan distance based RRT-A* is better than several other algorithms.

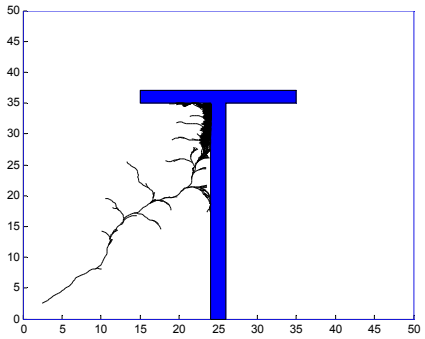


Fig. 6 Local minima problem of RRT-A* in "T" environment

The RRT-A* works effectively in the above environments, but does not in the environments which include local minima. Figure 6 is the car-liked model's planning curve generated by RRT-A* based on Manhattan distance function in the environment which includes "T" obstacles after 5000 iteration, at last, it failed to plan a path. For local minima problem, there are a lot of schemes, such as the mechanism of collision information detection in [10] and regression testing mechanism in [13].

5. CONCLUSION

The introduction of A* cost function makes the drawback of deficiency of cost function up. And it guided the tree to a better path. The heuristic information function can influence the performance of planner. Simulation results of several improved heuristic algorithms verify the excellent performance of the improved RRT algorithm, and there are various performances in the heuristic RRT algorithms based on different heuristic information functions.

REFERENCES

[1] H. F. Durrant-Whyte. Where am I? A tutorial on mobile vehicle localization. Industrial Robot, 1994, 23(2): 11-16.

[2] J. T. Schwartz and M. Sharir. On the piano mover's problem: I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers. Communications on Pure and Applied Math., 1983, 36: 345-398.

[3] J. P. Laumond. Finding collision-free smooth trajectories for a non-holonomic mobile robot. In Proc. Int. Joint Conf. on Artif, 1987:1120-1123.

[4] A. M. Shkel and V. J. Lumelsky. Incorporating body dynamics into sensor-based motion planning: The maximum turn strategy. IEEE Trans. Robot. Autom, 1997, 13(6): 873-880.

[5] S. M. Lavalle. Rapidly-exploring random trees: A new tool for path planning [R] . Iowa, USA: Computer Science Department, Iowa State University, 1998.

[6] C. Urmson and R. Simmons. Approaches for heuristically biasing RRT growth, in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. 2003, 1178-1183.

[7] P. E. Hart, N. J. Nilsson, et al. A formal basic for the heuristic determination of minimum Cost paths [J]. IEEE Transactions on Systems Science and Cybernetics, 1968, 4(2): 100-107.

[8] R. M. Murray, S. S. Sastry. Non-holonomic motion planning: steering using sinusoids [J]. IEEE Transactions on Automatic Control, 1993, 38(5): 700-716.

[9] R. M. Murray, Li ZX, S. S. Sastry. A mathematical introduction to robotic manipulation [M]. USA, Floraida: CRC Press, 1994.

[10] P. Cheng, S. M. LaValle. Reducing metric sensitivity in randomized trajectory design. In Proceeding of IEEE/RSJ International Conference on intelligent Robots and Systems, 2001, 43-48.

[11] J. Guitton, J. L. Farges and Raja Chatila. Cell-RRT: Decomposing the environment for better plan, in IEEE/RSJ International Conference on intelligent Robots and Systems, 2009, 5776-5781.

[12] L. Jaillet, J. Cortes and T. Simeon. Sampling-based path planning on configuration-space costmap, IEEE Transactions on Robotics. 2010, 26:647-659.

[13] M. Kalisiak, M. V. Panne. RRT-Blossom: RRT with a local flood-fill behavior, IEEE International Conference on Robotics and Automation. 2006, 1237-1242.