Collision Probability based Safe Path Planning for Mobile Robots in Changing Environments

Hong Liu[a], Chuangqi Wang[b]

Key Lab of Machine Perception and Intelligence

Shenzhen Graduate School, Peking University, China

[a]Hongliu@pku.edu.cn, [b]wangchuangqi@sz.pku.edu.cn

Keywords: Manipulator Planning, Man-Machine-Environment Engineering, Mechanism and Robot.

Abstract. Automatic path planning has many applications in robotics, computer-aided design(CAD) and industrial manipulation. The property of safety is vital but seldom taken into consideraion by typical path planning. In this paper, collision probability is introduced as an evaluation of crowd degree of environments to get a safer path. The smaller collision probability a node has, the more possibly the node can be extended. Meanwhile, the in/out degree of a node is limited to prevent some nodes to be extended excessively. Through evaluating collision probability on-line, a safe path planning based on DRRTs, called Safe-DRRT, is proposed to provide a path not only feasible but also safe. Finally, a path planner is implemented with Safe-DRRT as a guidance and a local planner. In plentifully crowded experiments with moving obstacles, the proposed method has demonstrated to be competent compared to the state of the art.

Introduction

In the field of Human-Robot Interaction (HRI), the property of safety is clearly vital. As long as humans and robots share the same workspace or cooperate in the same task within any environment, safe path planning must be considered seriously. Typical path planning [1] seldom pays attention to it. Although some works devote to solve it, safe path planning remains a challenge, especially for a robot of high degrees of freedoms (DOFs) in crowded and sharply changing environments.

After Lezano-Perez introduced the concept of configuration space (C-space) [1] into the context of planning, methods building an approximate representation of the C-space catch the focus of research. And based on C-space, sampling-based methods are introduced into path planning field. A lot of them [2,3,5-8] are very successful in solving planning problems for mobile robots even with high DOFs, which were previously considered intractable. However, majority of them just provide feasible paths that are only guaranteed collision-free, which are difficult to apply into pragmatic environments. For safe paths, a suitable safety property needs to be defined, such as keeping a distance from obstacles as much as possible[4], moving along the medial axis of the free-space [5] or defining a danger field [6,7], etc. A general framework for motion planning in HRI environment is proposed by Brock and Khatib [8] and our eariler work [9] also devotes to safe interaction between human arms and robot manipulators.

In addition, motion planning usually can be divided into two-stages: path planning and trajectory planning [1]. Path planning extracts a sequence of nodes that don't take dynamics, velocity and path's smoothing into consideration, which will be dealt with in trajectory planning. Therefore, path planning considering safety is also demanded which has seldom been researched, to guarantee more efficient trajectory planning.

In this paper, combining sample-based path planning and safety assessment, a method of Safe-DRRT is proposed to keep the extracted path as safe as possible in crowded and changing environments based on DRRT [10]. A path is extracted which is not only collision-free but also safe. For each node of the search tree rooted at the goal configuration, the crowd degree is evaluated as collision probability. When the extending procedure is invoked after the extending direction is generated, not the nearest node but the *k* nearest neighbors are selected. And after excluding nodes

whose degree exceed the defined threshold, the one with the smallest collision probability is chosen. When environments are changing, each node is reevaluated and a new solution is extracted.

The remainder of the paper is organized as follows. Related works are introduced in Section II. And estimation of collision probability is described in Section III. While the proposed planning algorithm that seeks for a safer path based on DRRTs is described in Section IV, experimental results are demonstrated and analyzed in Section V. Section VI draws final conclusions.

Related Works

Rapidly-exploring Random Trees (RRTs). As a famous single-query method, RRTs [3] is researched extensively since it's biased toward unexplored parts based on voronoi regions. The standard RRT algorithm grows a search tree from a start configuration q_{start} to a goal configuration q_{goal}. To grow the tree, an extended direction q_{rand} is randomly selected and then the nearest node $q_{nearest}$ in the tree is selected based on some metric. Finally, a new node q_{new} is extended with a certain step by growing the tree from $q_{nearest}$ to q_{rand} only if the edge between q_{new} and $q_{nearest}$ is collision free. This process is iterated until a solution is extracted or some user-defined threshold is reached. It will be more efficient if the extended direction is goal configuration with some probability p_g and is a random sample with probability $1\text{-}p_g$. In static environments, the standard RRT can explore the environments effectively and a solution is found rapidly. However, for dynamic environments, RRTs is not competent since much time is spent for growing a new tree from scratch.

DRRT. a replanning algorithm, DRRT[10] is proposed to regrow a new tree by just efficiently removing the invalid parts and maintaining the rest. When environments are changing, all the invalid parts of the exploring tree are trimmed and the remaining tree is regrown for a solution. Through utilizing the previous tree, DRRT generates a new solution efficiently. However, since the premise is that just a valid solution is extracted as rapidly as possible, the solution may be not effective in severely changing environments since the safe property is not considered.

Safe Path Planning. Some works have taken safety property into consideration. MAPRM [5] is proposed to retract sampled configurations onto the medial axis of the free space. Although it's effective in static environments, complicated geometric calculation makes it difficult to apply into dynamic environments. In [12], safe-RRT is introduced in uncertain environments by combining RRTs and EKF. A safe path is extracted through a large number of detected collisions. In addition, HNN and danger field are applied into safe path planning [6, 13]. In our works, all the extended nodes are used to evaluate collision probability of a node whether in collision or free, which utilizes nodes more efficiently, without increasing the number of collision detection.

Our Proposed Method: Safe-DRRT

Inspired by DRRT, a new path planner, named Safe-DRRT, is proposed by considering collision probability. Meanwhile, nodes' degree is limited to prevent the nodes to be extended excessively. A safe path is extracted through extending the tree from the sparsest one of k-nearest neighbors to q_{rand}. Next, evaluation of collision probability and details of our algorithm are introduced.

Evaluation of collision probability (CP)

For a configuration node, the collision probability is a integration of the probability colliding with all obstacles after a time δ_t. Obviously, the lower the collision probability of a node is, the safer the surrounding region is. Consequently, this node should be extended greatly and may have a higher probability to be a node of the path.

Firstly, we define a function that determines whether a node is in collision or free by $\delta_t(n,W)$:

$$\delta_t(\mathrm{n},W)=\begin{cases} 0, & n\in \textit{free space of } W \\ 1, & n\in \textit{collision space of } W \end{cases}. \tag{1}$$

here, in the time *t*, when the node *n* is in collision, $\delta_t(n,W)$ is set to 1. Otherwise, it is set to 0.

Inspired by [14], estimation of collision probability (CP) for a node after a time slot δ_t can be defined theoretically as:

$$\mathrm{CP}_{(t+\delta_t)}(n) = \int_W \delta_t(n,W) * P_t(W) dV_w \,. \tag{2}$$

where, $\delta_t(n,W)$ denotes whether the node *n* is in collision or free as defined by Eq. 1. And $P_t(W)$ represents the probability density of environments in the current time *t*. An integration variable, V_w, is selected by the volume unit of C-space.

Practically, there are lots of obstacles in the environment and they usually move independently. Therefore, the likelihood that the node *n* is in collision can be computed by the product of probabilities that *n* collides with all the obstacles:

$$\mathrm{CP}_{(t+\delta_t)}(n) = \prod_{i \in obstacles} p_t^{coll,i}(n) \,. \tag{3}$$

here, $P_t^{coll,i}(n)$ is the probability of colliding with the obstacle *i*. However, in dynamic environments, collision probability of a node is only influenced by obstacles within a certain range. And the remaining task to compute the probability that collides with an obstacle $P_t^{coll,i}(n)$ is still a challenge. Although some methods can be used to evaluate this likelihood such as the normalized distance from the obstacle, or Monte Carlo technique and etc, it's still difficult for these methods to apply since it requires many samples for an accurate estimation. And some computation is wasted for distance calculation since collision probability is just influenced by its surrounding region.

In the proposed method, Parzen Window Density is used to estimate the collision probability which is extensively used in classification, image registration and so on. Given an observation, Parzen-window estimates the Probability Density Function (PDF) from which the sample was derived. Firstly, a window function can be placed at a node *n* and be used to determine how many observations fall within the window and then the PDF value *P(n)* is the total contributions from the observations in this window. Therefore, collision probability of the node *n* can be shown as:

$$\mathrm{P}(n) = \mathrm{Max}\left(\frac{\sum_{window} \mathrm{N}_{\mathrm{invalid}}}{\sum_{window} (N_{valid} + N_{invalid})}, 0\right). \tag{4}$$

where, N_{valid} is the number of valid nodes while $N_{invalid}$ represents the number of invalid nodes. In addition, *P(n)* is set to 0 when no nodes exist in this window.

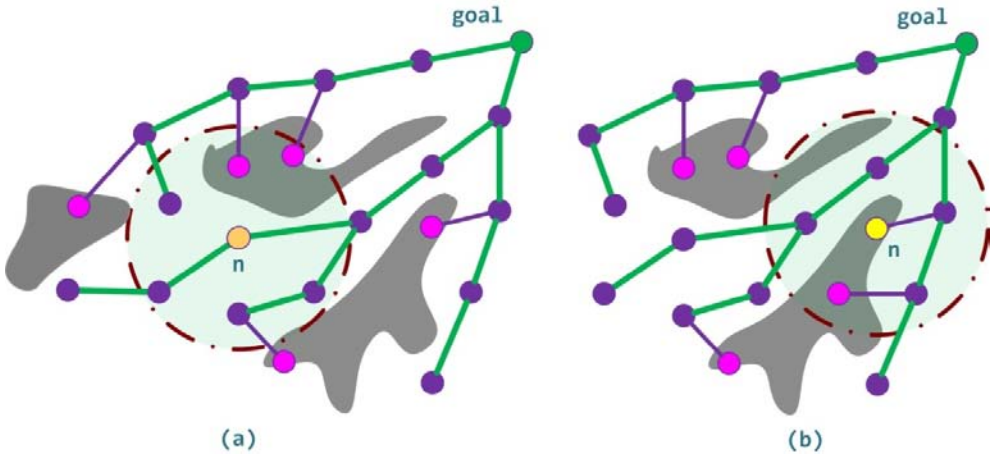


Fig. 1. collision probability is estimated based on Eq. 5. (a) shows collision probability of the valid node. (b) shows collision probability of the invalid node.

Moreover, since all the extended nodes are stored into the search tree, collision probability can distinguish valid nodes from invalid ones. Therefore, the collision probability is defined as:

$$\mathrm{CP}(n)=\begin{cases} p(n), & n \in valid\ \ nodes \\ p(n)+1, & n \in invalid\ \ nodes \end{cases}. \tag{5}$$

Obviously, *CP(n)* is [0, 1) for valid nodes and (1, 2] for invalid nodes. Fig. 1 shows the evaluated collision probability. Each node firstly estimates the PDF value *P(n)* based on k-nearest nodes and then *CP(n)* is estimated based on Eq. 5. For different types of nodes, evaluation is shown in Fig. 1(a) and (b).

Details of Safe-DRRT

Based on collision probability, Safe-DRRT is introduced. The details are shown in Algorithm-1. And the general process is illustrated in Fig. 2.

Node Extension. For each node, not only collision detection is executed to identify the node whether in obstacles or not, but also collision probability is evaluated for a safer path, which has been shown in Section III. In addition, in order to ensure that difficult region can be explored by the search tree, in/out degree of each node is considered to limit nodes' extending. When the degree of a node has reached a predefined threshold, it will be not considered even if it is the safest one of k-nearest neighbors. Therefore, the node should be selected to be extended based on Eq. 6.

$$q_{near}=\{i \mid \mathrm{Min}(\mathrm{CP}(i)) \,\&\&\, \mathrm{Degree}(i) < Threshold,\quad i \in \{1,2,3,...,k\}\}. \tag{6}$$

here, *Degree*(*i*) represents degree of the node *i*. Moreover, the number of neighbors is denoted by *k*.

Algorithm 1: Safe-DRRT

Input: Max iteration N, Growing Tree T

```
1  T.Init(q_goal);
2  GrowRRT(P_safe, T);
3  while not reach q_goal do
4      check the P_safe and move the robot along it;
5      if sub P_safe is in collision then
6          UpdateNode(T);
7          GrowRRT(P_safe, T);
8      end
9  end

10 GrowRRT(P_safe, T)
11 if P_safe doesn't exist then
12     while k≤Max_iteration and GapSatisfied() do
13         q_rand = ChooseState(P_b);
14         L_Near = NearestNeighbors(q_rand);
15         q_near = ChooseSafe(L_Near);
16         q_new = ExtendSafe(q_near, q_rand, Δ_q);
17         q_new.cp = Mark_CP(q_new);
18         T.add(q_new);
19     end
20 end

21 UpdateNodes(T)
22 for q_k in T do
23     if OutOfRange(q_k) then
24         TrimsubTree(q_k);
25     end
26     else
27         if IsCollision(q_k) then
28             q_parent = q_k.GetParent();
29             P_safe.Invalid(q_k);
30             CreateTreeFromNode(q_parent);
31         end
32         else
33             q_new.cp = Mark_CP(q_new);
34         end
35     end
36 end
```

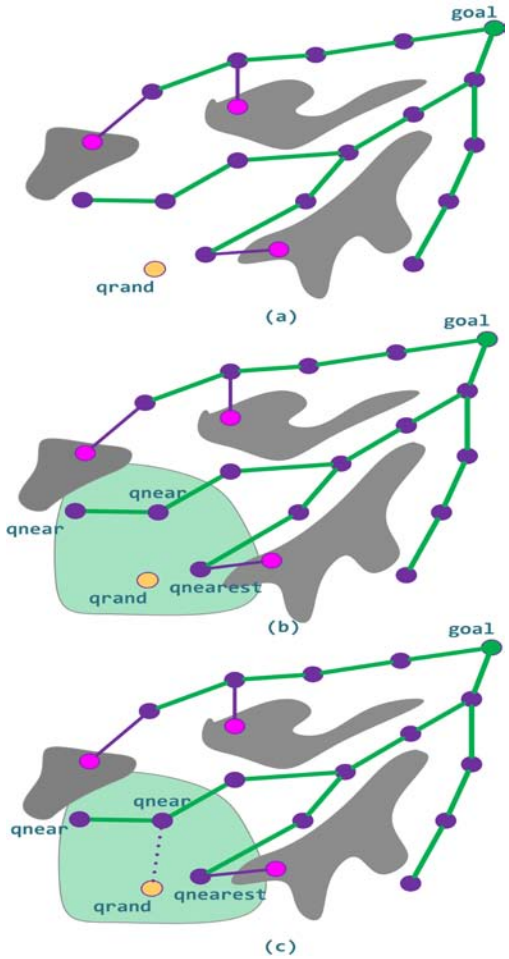


Fig. 2. Node extending. (a) q_{rand} is chosen on an incremental search tree. (b) k-nearest neighbors are selected. (c) based on collision probability, the safest node is selected and extended.

Algorithm of Safe-DRRT. In algorithm 1, Firstly, a search tree is initialized and rooted at q_{goal} and *GrowRRT()*(line 10-20) is executed to build an initial tree *T*. In the *GrowRRT()*, the extended direction q_{rand} is generated with goal bias (see Fig. 2(a)). The k-nearest neighbors L_{near} are selected in the tree closest to q_{rand} (see Fig. 2(b)). Then after q_{near} is chosen based on Eq. 6, q_{new} is extended and its collision probability is evaluated based on Eq. 5 (see Fig. 2(c)). Secondly, execution and tree's updating are operated iteratively until q_{goal} is reached. In the execution, guided by the extracted path P_{safe}, a local trajectory Tra_{safe} is planned by the local planner and the robot keeps moving along the trajectory if the next node of Tra_{safe} is collision-free. When the trajectory Tra_{safe} is in collision, information will be feedback to Safe-DRRT and the search tree is updated by the procedure *UpdateNode(T)*(line 21-36) followed by growing the existed tree if necessary. In the *UpdateNode(T)*, since mobile agent has moved to a new location, each node is judged whether in the region of new environments or not. If not, the subtree rooted at this node will be trimmed (line 24). Then if the node is in collision, it will be marked as invalid and weighted like Fig. 1 (b) and then several nodes can be grown from its parent node (line 27-30) in order to make the tree cover C-space better. Otherwise, it will be reevaluated (line 33) like Fig. 1(a).

Experiments and Analysis

In order to evaluate the proposed method, hundreds of simulation experiments are implemented in 3D workspace with different scenarios. Safe-DRRT is introduced into a two-level framework to guide a local path planner which is local RRTs in our experiments. Moreover, success rate and the average time cost for a task can be used to evaluate the safe property.

To testify the validation of our proposed method, the experiments are performed on the models of a 3-DOF vehicle and a 9-DOF mobile manipulator. A manipulator for 6-DOF Kavasaki FS03N, which is mounted on a mobile base, is simulated and shown in Fig. 3. All of our experiments are carried out on a personal computer with a processor 2.71 GHz with 2 GByte of memory. The planner is implemented in C++ and the collision checks are used with PQP 1.3, an open-source 3D collision detection library. The k-nearest neighbors are searched with open source kd-trees.

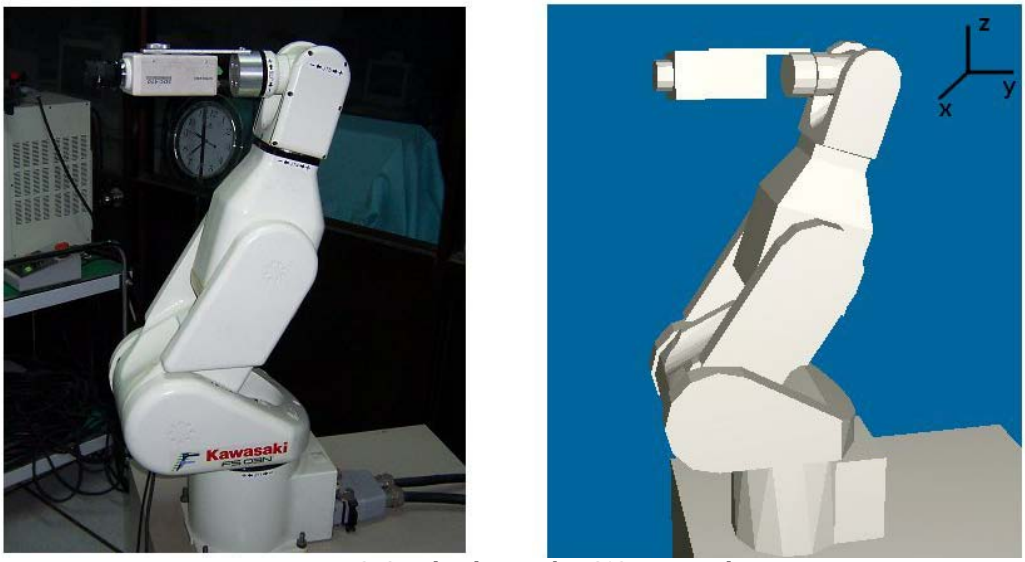
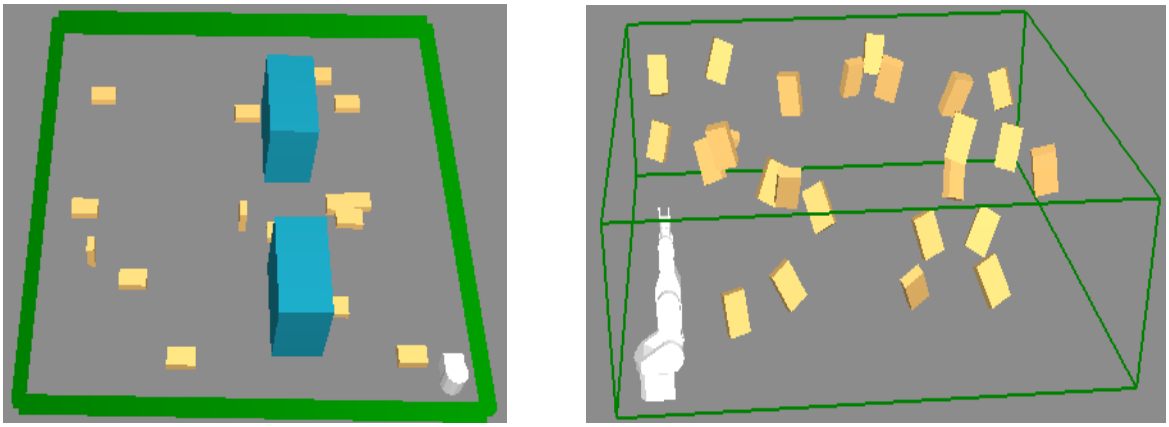


Fig. 3. Simulated Kavasiki FS03N Manipulator

We ascribe these experiments into two groups with moving obstacles which translate and rotate by a random step with time. The only information we are assuming the agent has access to during planning is the position information of dynamic obstacles. Fig. 3 demonstrates the scenarios of designed environments.

The first group of experiments is a 3-DOF vehicle among some changing/stationary obstacles. All these moving objects are supposed to move on the ground. In group II, the algorithm is evaluated with our simulated 9 DOF mobile manipulator in a more complicated environment.



(a) group I (b) group II

Fig. 4. The scenario of experiment group I and II

Settings of these experimental environments are listed in Table I. Here, r, o_i, s and n_o represent the robot size, size of the i_{th} obstacle, scene size and the number of obstacles respectively. The size shown in Table I is the AABB boundbox of obstacles conveniently.

Table I. Different scenarios of experiments

Sizes	*Group I*	*Group II*
r	30 × 20 × 10	30 × 20 × 90
o_1	65 × 35 × 10	—
o_2	12 × 15 × 6	—
o_3	—	10 × 15 × 10
s	300 × 300 × 10	300 × 300 × 90
n_0	2, 15	25

Motion of unpredictable obstacles is defined by six parameters indicating random translation steps and rotation steps around three Cartesian coordinates. The walk steps are randomly chosen in (-2*cm*, 2*cm*) while the random rotation steps in (-15°, 15°). Meanwhile the initial positions of obstacles are randomly set.

For each experiment, planning is deemed to fail if the mobile robot can't reach its goal area when time threshold is reached. In our experiments, the time threshold is 40 seconds for group I and 60 seconds for group II. Moreover, the number of nearest neighbors, k, is set to 6.

Results of our approach comparing with DRRT are shown in Table II together. The column of T_{avg} represents the average time per replanning, while R_{success} denotes the success rate to its goal area in a limited time interval. T_{update}, T_{regrow} represent the average time of *UpdateNodes()* and *GrowRRT()* separately, while T_{lplan} denotes the cost time for local planner which does not exist in DRRT. Results show that performance of our proposed method is obviously superior to DRRT.

For our method, several reasons mainly attribute to a good performance. Firstly, all the extended nodes are updated even for invalid nodes since these nodes can be free for next iteration and can guide the tree to grow far away from them as the footstone. Therefore, The cost time of the procedure *GrowRRT()*, is far less than that of DRRT, since invalid nodes can become valid for a new solution(see col. 4 in Table II).

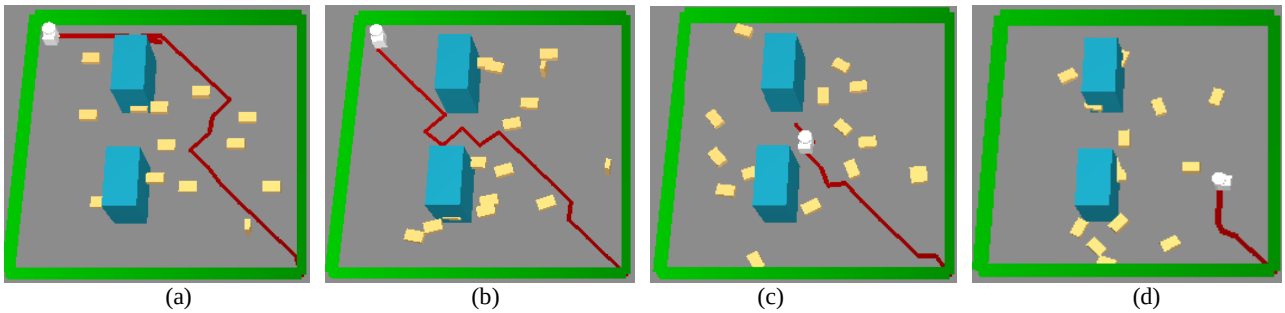
Secondly, collision probability is evaluated which makes the search tree grow to the free area as much as possible. However, in some situations, the tree must grow towards difficult regions such as narrow passages. In order to force the tree to extend to these difficult regions, the degree of each neighbor is considered when q_{near} is selected from the k nearest neighbors. Finally, collision probability (CP) integrated with limiting degree of nodes makes sure that free regions can be taken priority to be extended than difficult regions and difficult regions can also be extended if necessary. And all the free regions can be explored more reasonably. Consequently, under the guidance of Safe-DRRT, the success rate compared with DRRT has been greatly improved according to the Table II (see col. 7 in Table II).

Table II. Experimental results compared with DRRT

Group	Method	T_{update}	T_{regrow}	T_{lplan}	T_{avg}	$R_{success}$
I	DRRT	0.3957	0.4757	—	0.8743	28%
I	our method	0.3701	0.0561	0.2016	0.6277	82%
II	DRRT	0.3887	0.5886	—	0.9774	20%
II	our method	0.3952	0.0391	0.3134	0.7477	89%

Note that the gap *0.0001* is due to rounding.

In Fig. 5, several snapped pictures in group I are provided since it is simpler and more intuitive than group II. The initial path extracted by our path planner is shown in Fig. 5(a) and (b). The path is updated after several iterations if necessary and the mobile robot can move guided by this updated path (See Fig. 5(c)). Finally, the robot is directed to move to the goal position (See Fig. 5(d)). It can be seen that the path is far from obstacles and walls marked in green in any possible way. Therefore, regions that the path passes through can keep collision-free for a longer time and then it's safer. In addition, these pictures can show that smoothing the path is easy for trajectory planning and consequently, it's reasonable that our method can be applied into pragmatic HRI environments in future.



(a) (b) (c) (d)

Fig. 5. Snapped pictures of group I

Conclusions

In this paper, a method of Safe-DRRT is proposed to apply into HRI environments. By considering the collision probability and degree of each node to limit some nodes growing excessively, a search tree is generated and a safer path is extracted. When environments are changing, the proposed method can update the information of nodes and generate a new path which is well qualified for environments with dynamic obstacles. Since a safe path is extracted, the trajectory is easier to generate by the local planner. Therefore, combined with some local path planners, our method can be easily extended to pragmatic environments, which is our main further work.

Acknowledgements

The works was supported by National Natural Science Foundation of China (NSFC, No.60875050, 60675025), National High Technology Research and Development Program of China (863 Program, No.2006AA04Z247), Shenzhen Scientific and Technological Plan and Basic Research Program (No. JC200903160369A), Natural Science Foundation of Guangdong (No. 9151806001000025).

References

[1] S. M. LaValle, "Planning Algorithm", Cambridge University Press, 2006.

[2] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "*Probabilistic roadmaps for fast path planning in high-dimensional configuratin spaces*", IEEE Transactions on Robotics and Automation, vol. 12, pp. 566-580, 1996..

[3] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects", Algorithmic and Computational Robotics: New Directions, pp. 293-308, 2000.

[4] J. Van den Berg and M. H. Overmars, "Planning the shortest safe path amidst unpredictably moving obstacles" Proc. Int. Workshop on Algorithmic Foundation of Robotics (WAFR), pp.885-897, 2006.

[5] S. A. Wilmarth, N. M. Amato and P. F. Stiller, "MAPRM: A Probabilistic Roadmap Planner with Sampling on the Medial Axis of the Free Space", IEEE International Conference on Robotics and Automation (ICRA), pp. 1024-1031, 1999.

[6] B. Lacevic, "Sampling-based safe path planning for robotic manipulators", IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pp.1-7, 2010.

[7] B. Lacevic and P. Rocco, "Towards a Complete Safe Path Planning for Robotic Manipulators", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5366-5371, 2010.

[8] H. Liu, X. Deng and H. Zha, "A Planning Mehod for safe Interaction between Human Arms and Robot Manipulators", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp.1814-1820, 2005.

[9] O. Brock and O. Khatib, "*Elastic Strips: A Framework for Motion Generation in Human Environments*", International Journal of Robotics Reasearch (IJRR), Vol. 21, No.12, pp. 1031-1052, 2002.

[10] D. Ferguson, N. Kalra and A. Stentz, "Replanning with RRTs", IEEE International Conference on Robotics and Automation (ICRA), pp. 1243-1248, 2006.

[11] D. Ferguson, A. Stentz, "Anytime, Dynamic Planning in High-dimensional Search Spaces", IEEE International Conference on Robotics and Automation (ICRA), pp. 1310-1315, 2007.

[12] R. Pepy and A. Lambert, "Safe Path Planning in an Uncertain-Configuration Space using RRT", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp.5376-5381, 2006.

[13] W. Chen, C. Fan and Y. Xi, "On-Line Safe Path Planning in Unknown Environments", IEEE International Conference on Robotics and Automation (ICRA), pp. 4191-4196, 2003.

[14] P. E. Missiuro, N. Royi, "Adapting Probabilistic Roadmaps to Handle Uncertain Maps", IEEE International Conference on Robotics and Automation (ICRA), pp. 1261-1267, 2006.