Proceedings of the IEEE
International Conference on Automation and Logistics
Qingdao, China September 2008

Mobile Robot Path Planning Based on an Improved Rapidly-exploring Random Tree in Unknown Environment*

Liu Chang-an, Chang Jin-gang, Li Guo-dong, Liu Chun-yang
School of Computer Science and Technology
North China Electric Power University
Beijing, 102206 China
liuanhit@163.com

Abstract - **To deal with the path planning of mobile robot in unknown environment, an improved rapidly-exploring random tree (RRT) is presented by analyzing basic RRT algorithm. We applied improved RRT algorithm to local navigation of a mobile robot in unknown environment. Detecting local unknown environments by robot sensor system, the method orients a biased direction from RRT algorithm is used to drive the robot to avoid obstacles and move to the goal. The method is not only simple and flexible, but also reduces the nodes of tree and the amount of calculation for real-time computation. We conducted the simulation to verify the proposed method in real unknown environment.**

Index Terms – Mobile Robot, Path Planning, Rapidly-exploring Random Tree, Biased Direction

I. INTRODUCTION

Intelligent mobile robot is a robot system which can perceive its environments and state through sensors and autonomously move to the goal (navigation) in the environments with obstacles so as to accomplish special task. Navigation technology of autonomous mobile robot in unknown environment has become the main technology of space detection robot. Navigation of autonomous mobile robot which can adapt to unknown environment should possess the capability of environments cognition, behaviour decision-making, movement controlling etc. The main research of this field includes system structure, circumstance modelling and localization, path planning, movement controlling, fault diagnosis and fault tolerant controlling etc.

The path planning of mobile robot in an unknown environment is the core technology in mobile robot research, which attracts a large number of researchers all over the world. It refers to find a path from the start point to the goal point in the work environment with unknown obstacles, and assures the safe, non-collisions and avoidance of all obstacles as mobile walking. According to the degree of integrity of the environmental information, robot path planning can be divided into local path planning and global path planning[1][2]. There are considerable researches on the first field, such as space-based partition of unity method, projection method, configuration space method and so on. In local planning, environment information is unknown, or is only partly known, and robot sensor system can only detect local environmental information it perceived. The most representative study in this field is artificial potential field (APF)[3]and probabilistic roadmap (PRM)[4]. Artificial potential field is simple and innovative, which has been widely used in recent years, but is can easily lead to a partial die point, which makes the robots unable to go to the goal. Probabilistic roadmap method in complicated environments is difficult.

In this path planning field one approach has been much concerned: Rapidly-exploring Random Tree (RRT)[5][6], which is based on stochastic search strategies. RRT is a data structure and algorithm that is designed for efficiently searching unknown environment and non convex high-dimensional spaces. RRT are constructed incrementally in a way that quickly reduces the expected distance of a randomly-chosen point to the tree. RRT are particularly suited for path planning problems that involve obstacles and differential constraints (nonholonomic or kinodynamic). RRT can be considered as a technique for generating open-loop trajectories for nonlinear systems with state constraints. An RRT can be intuitively considered as a Monte-Carlo way of biasing search into largest Voronoi regions. Some variations can be considered as stochastic fractals. Usually, an RRT alone is insufficient to solve a planning problem. Thus, it can be considered as a component that can be incorporated into the development of a variety of different planning algorithms.

The RRT approach to path planning introduces a technique of determining a planning path by selecting random points within a known environment and moving towards that point an incremental distance from the nearest node of an expanding tree. The movement from existing traversed points to random points in the environment will make a path that looks like a tree, and will cover most of the free space in the environment. A planned path will be found when a branch in the tree comes close the goal positions.

* This work is partially supported by the National Natural Science Foundation of China (60775058) and the key Project of China Ministry of Education (107028).

978-1-4244-2503-7/08/$20.00 © 2008 IEEE

The RRT algorithm has advantages at exploring free space in large and unknown environments as well as parallelizable on unlike many other path-finding algorithms. However, the termination condition of tree formation may be limited by the success in finding goal. This condition may result in the tree containing many nodes in the order of hundred. Due to these drawbacks, there are some strategies that can help the original RRT algorithm to reduce the number of nodes. The number of node is an important factor in computation to find an optimal path using the tree[7][8].

In this work we consider an improvement to the RRT algorithm, one bias-direction parameter was added when the tree is randomly set up to reduce the nodes and the amount of calculation is decreased accordingly. We conducted the simulation to certificate the effectiveness and feasibility of the method based on unknown environment.

II. BASIC RRT ALGORITHM

The Rapidly-exploring Random Tree (RRT) was introduced in[9] as an efficient data structure and sampling scheme to quickly search high-dimensional spaces that have both algebraic constraints (arising from obstacles) and differential constraints (arising from nonholonomy and dynamics).

The basic RRT construction algorithm relies on a tree structure being built that contains the start point as its root node, and eventually, the goal point as one of its leaves. The construction of the tree revolves around picking random points in the environment, finding the nearest point in the tree to this random point, then moving towards that point an incremental distance. If the incremental movement does not encounter an obstacle, insert the point as a new node in the tree. Eventually, the newly inserted node will be close to the goal node. Once this occurs, a path from goal to start point can be made by traversing up the tree until the root node is reached.

Agorithm 1:

BUILD_RRT(x_{init})

1 $T.init(x_{init})$

2 do for k=1 to K do

3 $x_{rand} \leftarrow$ RANDOM_CONFIG( ) ;

4 EXTEND(T, x_{rand}) ;

5 Return T

EXTEND(T, x)

1 $x_{near} \leftarrow$ NEAREST_NEIGHBOR(T, x)

2 if NEW_CONFIG(x, x_{near}, x_{new}) then

3 $T.add\_vertex(x_{new})$;

4 $T.add\_edge(x_{near}, x_{new})$;

5 if $x_{new} = x$ then

6 Return *Reached* ;

7 esle

8 Return *Advanced* ;

9 Return *Trapped* .

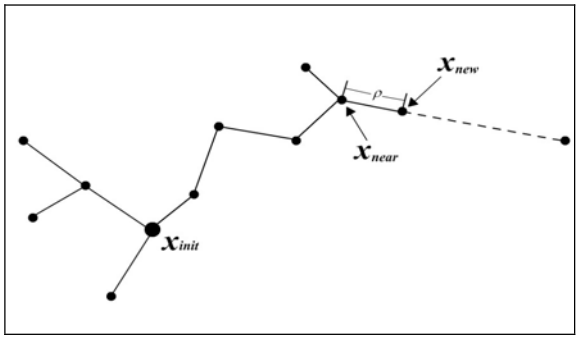

Fig. 1 The EXTEND operation.

The basic RRT construction algorithm is give in Algorithms 1. A simple iteration in performed in which each step attempt to extend the RRT by adding a new vertex that is biased by a randomly-selected configuration. The EXTEND function, illustrated in Fig.1, selects the nearest vertex already in the RRT to the given sample configuration, x. The function NEW_CONFIG makes a motion toward x with some fixed incremental distance ρ, and tests for collision. This can be performed quickly ("almost constant time") using incremental distance computation algorithms. There situations can occur: *Reached*, in which x is directly added to the RRT because it already contains a vertex within ρ of x; *Advanced*, in which a new vertex $x_{new} \neq x$ is added to the RRT; *Trapped*, in which the proposed new vertex is rejected because it does not lie in C_{free}.

III. THE IMPROVED RRT ALGORITHM

We are considering RAND_CONFIG function. According basic RRT algorithms, RRT can be used in isolation as a path planner because its vertices will eventually cover a connected component of free space, coming arbitrarily close to any specified x_{goal}. The problem is that tree is extended random toward any directions, without any bias toward the goal, convergence might be slow. So extend tree cover with all planning space, however, path only actually need a part of extend tree form start node to goal node. The idea is to improve the efficiency of the method by setting angle of deviation parameter, extend tree can randomly generate configurations according to prescriptive direction towards unexplored areas. According to basic RRT algorithm, the construction of the tree revolves around picking random points in the environment, finding the nearest point in the tree to this random point, then moving towards that point an incremental distance. If the incremental movement encounters an obstacle, this incremental movement will be

unprofitable, and the calculation of finding the nearest point in the tree to this random point also will be excrescent. So we add the direction parameter in picking the random points. Instead of moving an incremental distance to a random point just once, keep moving towards it until it is reached, or an obstacle is hit. When the robot find an obstacle, the range of the direction of the tree picking a random point will decrease a t this position.

Determining the direction parameter is the core technology in improving RRT algorithms. In this paper, the direction parameter is related with not only shape of obstacle and sensors' detecting range, but also position of the goal. We consider obstacle as a protruding. According the demand of path planning and character of obstacle model, we use 2D laser detecting radar as primary sensor. Because environment was detected sensor system based on scan line can detect environments according to angle resolution. The detecting range of sensor system is give in Fig.2, d_f denote line direction towards goal, d_l denotes omni-directional in left rector area, and d_r denotes omni-directional in right rector area.

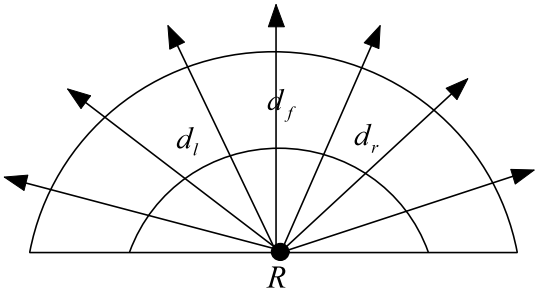

Fig. 2 The detecting range of sensor system.

Assume robot forward-motion all the time, we can use semicircular area detected by laser detecting radar as local path planning window. In the local window, d_f is determined according goal and current node of extended tree. Three instances came into question in the follow.

(1)Obstacles weren't detected by laser radar along d_f direction. Whether or not detect obstacle in left and right, random tree extends toward to goal along d_f direction. $\theta_{rand} = 0$, showed as follows in Fig.3.

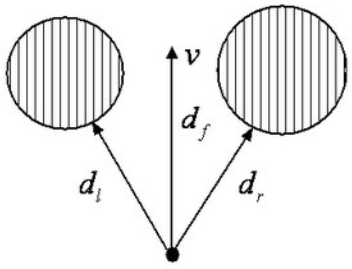

Fig. 3 Robot doesn't detect obstacle along d_f.

(2)Obstacles were detected by laser radar along d_f direction. Taking d_f as reference line, $\beta_l = \angle(d_l, d_f)$ is the angle between d_l and d_f, $\beta_r = \angle(d_f, d_r)$ is the angle between d_f and d_r. When $\beta_l < \beta_r$, random tree extends toward to goal along d_l direction, $\theta_{rand} = d_l$, showed as follows in Fig.4(a). When $\beta_r < \beta_l$, random tree extends toward to goal along d_r direction, $\theta_{rand} = d_r$, showed as follows in Fig.4(b).

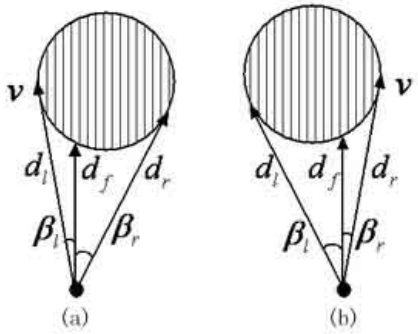

Fig. 4 Robot detects obstacle along d_f.

(3)When $\beta_l = \beta_r$ (special case of instance 2), random tree extends toward to goal along d_l direction, $\theta_{rand} = d_l$, showed as follows in Fig.5.

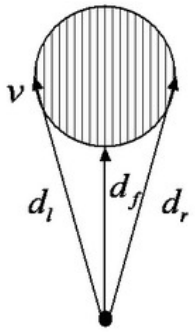

Fig. 5 special case of detecting obstacle along d_f

Avoid combining SI and CGS units, such as current in amperes and magnetic field in oversteps. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity that you use in an equation.

A function New RAND_CONFIG is used to replace the common RAND_CONFIG, showed as follows in algorithm 2.

Agorithm 2:

New RAND_CONFIG ( ): (when the robot has found an obstacle)

1 $x_{current} \leftarrow x_{new}$ (get from the last loop);

2 β_l , β_r ← According to the bound of obstacle and the ability of the sensor's exploration;

3 $\theta_{explored}$ ← According to the explored environments space;

4 $\theta_{feasible}$ ← Taking out the direction β_l, β_r and $\theta_{explored}$;

5 θ_{rand} ← Picking the exploring direction randomly in the $\theta_{feasible}$;

6 x_{rand} ← Picking the random point in the direction θ_{rand}.

Improved RRT algorithm is presented by using these function, it can meet demands of local obstacle avoidance, and also include the global information-the goal. Therefore, the algorithm is more efficient by combining local planning and global planning. By improving the basic RRT, it attempts to become effective to shorten the time of eyeless randomly exploring, and also reduces the memory of storing nodes of the whole tree.

IV. Simulations

Simulation of the improved RRT algorithm is used to predict the feasibility of the success of the improvement of basic RRT method.

In order to accomplish the RRT path planning simulation, there are three functions, which are very important to be built: Extend function: Extend (environment, current state, and target); Distance function: Distance (current state, target); Random State function.

Extend function is use to gain a new point, and the distance between this point to the certain point is certain. These new points generally move toward to the goal position. If the robot find some obstacle in the way toward the target, the program will make the extend function returned default value, EmptyStae, and also the program will be not succeed for distance estimation by repeat using extend function before the robot reaches the goal point. RnadomState function returns the value of the point picking from the work space randomly.

Path planning of in workspace with an obstacle is shown Fig.6, path planning of two obstacles is shown Fig.7, and path planning of many obstacle is shown Fig.8. Blue semi-circular is the detecting range of sensor system, centre of a circle is the position of robot. We describe planning process by path planning of an obstacle, as shown in Fig.6. Robot starts from *S*, and move to *G* because sensor system don't detect obstacle. When robot moves to position of Fig.6(a), sensor system detects obstacle in the forward direction. Extend direction is determined by improved RRT algorithm, so robot move and avoid obstacle, as shown in Fig.6(b). When robot moves to position of Fig.6(c), sensor system of robot doesn't detect obstacle in the path forward to *G*, so robot will move and arrive to *G*, as shown in Fig.6(d).

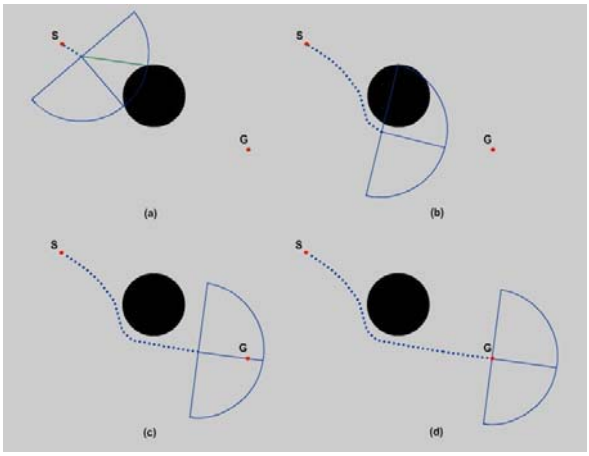

Fig. 6 Path planning in workspace with an obstacle.

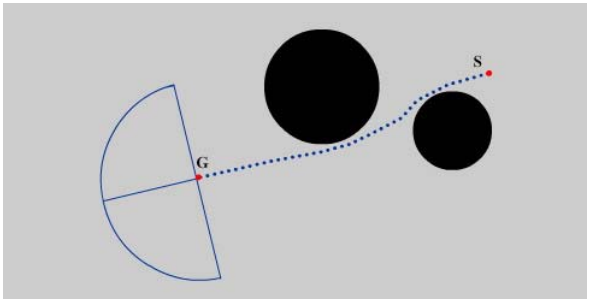

Fig. 7 Path planning in workspace with two obstacles.

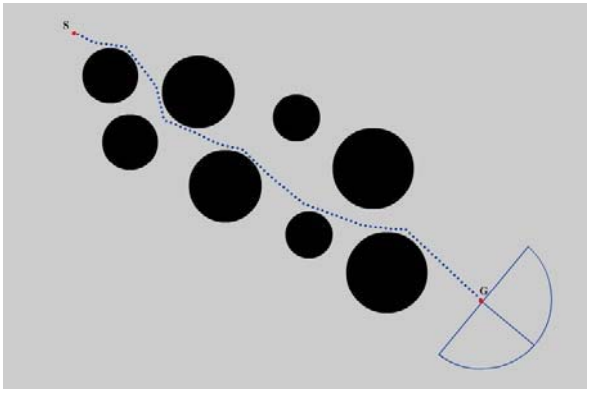

Fig. 8 Path planning in workspace with many obstacles

To allow a performance comparison between the two exploration strategies, we have run the same simulations of the two methods in workspace with many obstacles. According to the result, the program calculates the time of building the tree structure and the number of the node including of this structure.

Results obtained for the Basic RRT method and the Improved RRT method are shown in the Table I that the time which is used with the fore is much more than the latter one. The time can be saved, because when the robot finds some obstacles in traveling , it will spend less time than before for selecting the direction and decide which way could move on. By analyzing these data, the advantage of advanced RRT algorithm can be proved.

Table I The comparison of two algorithms of simulation

Trial	Basic RRT		Improved RRT	
	Time(s)	node	Time(s)	node
1	11.6	1375	9.2	1058
2	10.5	1257	10.1	1103
3	12.4	1389	12.2	1314
4	9.4	1024	9.1	987
5	14.3	1583	13.4	1428
Average	11.6	1326	10.8	1178

V. CONCLUSION

In this paper, we analyzed the basic RRT algorithm and proposed the modification method by adding bias parameter in determining direction when the tree is randomly set up. Improved RRT algorithm is applied to local navigation of a mobile robot in unknown environment, which can meet demands of local obstacle avoidance. Therefore, the algorithm is more efficient by combining local planning and global planning. We conducted simulations in unknown environments. From the results, we can conclude that the improved algorithm can reduce the number of nodes and the burden of calculation is decreased accordingly, so has broad application prospect.

REFERENCES

[1] Cai Zi xing, He Han gen, Chen hong. Some issues for mobile robots navigation under unknown environments [J], Control and Decision. Vol.8, no.4, pp.385-390, 2002.

[2] Latombe J C. Robot motion planning[M]. Boston, Kluwer Academic Publishers, 1991.

[3] Stuart Russell, Peter Norvig. Artificial Intelligence: A Modern Approach[M]. Prentice Hall, 2003.

[4] Lydia E Kavrakietal. Analysis of Probabilistic Roadmap for Path Planning[J]. IEEE Transactions on Robotics and Automation, vol.14, no.1, pp.166-171, 1998.

[5] J.J.Kuffner and S.M.LaValle. RRT-connect: An efficient approach to single-query path planning[C]. In Proceeding IEEE International Conference on Robotics and Automation, pp.995-1001, 2000.

[6] Bruce, J.Veloso, M. Real-time randomized path planning for robot navigation[C]. IEEE/RSJ Internatinal Conference on Intelligent Robots and System, vol,3, pp.2383-2388, 2002.

[7] P.Cheng, Z.Shen, and S.M.LaValle. Using randomization to find and optimize feasible trajectories for nonlinear systems[C]. In Proceedings Annual Allerton Conference on Communications, Control Computing, pp.926-935, 2000.

[8] Kamio, S., Iba, H. Cooperative Object Transport with Humanoid Robots using RRT Path Planning and Re-Plannnig[C]. IEEE/RSJ International Conference Intelligent Robots and Systems, pp.2608-2613, 2006.

[9] S.M.LaValle. Rapidly-exploring random trees: A new tool for path planning. TR98-11, Computer Science Dept, Iowa State University, Oct, 1998.