2021 IEEE 6th International Conference on Intelligent Computing and Signal Processing (ICSP 2021)

Implementation of SLAM and path planning for mobile robots under ROS framework

2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP) | 978-1-6654-0413-6/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICSP51882.2021.9408882

Zixiang Liu*
Shanghai University
Shanghai, China
liuzixiang3392@163.com

Abstract: **This paper mainly discusses the realization of real-time positioning and map construction of mobile robots, and the use of the improved artificial potential field method for local path planning. The FastSLAM algorithm based on particle filter is used to realize map construction. The autonomous positioning, path planning, dynamic obstacle avoidance and motion simulation of the mobile robot based on the ROS platform are realized on the gazebo platform.**

Keywords-ROS; gazebo; mobile; robot navigation; improved artificial potential field

I. PERFACE

With the rapid development of robot intelligence technology, the application of mobile robots in logistics support has become more and more extensive. In the future, when facing complex and unknown environments, robots need to accomplish three tasks: autonomous positioning, mapping and path planning. The first two tasks are instantaneous localization and mapping (simultaneous localization and mapping, SLAM) [1] of mobile robots, which are described as: in an unknown environment, the robot starts to move from an unknown position, and feedback data through sensors during the movement generate real-time state estimation to complete autonomous positioning, and at the same time build an incremental map. The third task is path planning. Its goal is to quickly find a collision-free motion path in the workspace with obstacles, so that the robot can bypass the obstacles and reach the target point.

In this paper, the FastSLAM [2-3] algorithm based on particle filter is used for map construction and it is simulated on the ROS platform. The artificial potential field algorithm is improved to solve the problem that the mobile smart car may collide with obstacles effectively when it is too far away from the target point. In the simulation, artificial potential field algorithm is used for local obstacle avoidance. Simulation proves that the ROS platform can meet the requirements of mobile robots for map construction, autonomous positioning, navigation and obstacle avoidance. In addition, the processing of sensor data by the ROS system also brings convenience to research.

II. SLAM

For the SLAM problem of mobile robots, t is used to represent time. X_t represents the pose of the robot. This article uses lidar as the main sensor, and the collected environmental information is two-dimensional, so X_t is usually a three-dimensional vector consisting of two-dimensional plane position coordinates and a rotation angle indicating the orientation. Over time, the pose sequence or trajectory can be expressed as $X_t=\{x_0,x_1,\cdots,x_t\}$, where x_0 (the initial pose) is known, and the other poses are temporarily unknown. u_t is used to represent the movement mileage from time t-1 to time t. Then the sequence $U_t=\{u_0,u_1,\cdots,u_t\}$ describes the relative movement of the robot. m is used to represent a map of the surrounding environment of the robot, which is composed of landmarks and objects, m describes their poses. It is usually assumed that the environment map is static. The robot measurement establishes the relationship between the features in m and the robot pose x_t. and the measurement sequence is expressed as $Z_t = \{z_1, z_2, \cdots, z_t\}$. The problem of SLAM is to recurrent the environment model m and robot pose sequence X_t based on the odometer and measurement data. As mentioned above, the SLAM problem requires the establishment of two mathematical models:

(1) A mathematical model describing the relationship between the odometer measurement and the robot's pose; (2) a model describing the relationship between the measurement value and the environment and the robot's pose. These models are usually regarded as probability distributions: $P(x_t|x_{t-1}, u_t)$ represents the probability distribution of the robot's pose x_t when the robot starts from a known pose x_{t-1} and the measured mileage data is u_t. $P=(z_t|x_t,m)$ represents the probability distribution of z_t measured at a known pose x_t in a known environment m. Then the probability distribution of hidden variables can be recurrent based on the measured data by Bayes rule.

The particle filter algorithm was first proposed by Gordon and Salmond in 1993, a new Bootstrap nonlinear filtering measure based on sequential importance sampling (SIS) method, which laid the foundation of the particle filter algorithm since then. It is a statistical filtering algorithm that combines Monte Carlo positioning and Bayesian estimation, also known as the sequential Monte Carlo method.

The particle filter algorithm mainly includes 4 stages: (1) The particle initialization stage, which generates a set of random samples in the environmental state space according to the empirical distribution of the system state vector; (2) The state transition stage, the prior probability of passing through the previous particle state to estimate the posterior probability density in the current environment state; (3) The weight calculation stage, in this stage, each particle will be re-scored.

Only particles with higher scores can generate new particles and can calculate the state of the next frame. Particles with low scores are eliminated. The specific scoring rules vary according to requirements. (4) The re-sampling stage, in this stage, particles with lower weights are eliminated, and particles with higher weights generate more particles and resample, which makes the algorithm converge towards the place with high weight. The FastSLAM algorithm based on the particle filter regards each particle as a specific estimation of the true value of the state. At any time, the FastSLAM algorithm maintains K particles in the following form:

$$X_t^{[k]}, \mu_{t,1}^{[k]}, \cdots, \mu_{t,N}^{[k]}, \Sigma_{t,1}^{[k]}, \cdots, \Sigma_{t,N}^{[k]} \tag{2-1}$$

Where [k] is the particle sample number, each particle contains a sample trajectory $x_t^{[k]}$ and N two-dimensional Gaussian sets with mean $\mu_{t,N}^{[k]}$ and variance $\Sigma_{t,N}^{[k]}$, where $n(1 \leq n \leq N)$ is the road sign number. k particles has k trajectory samples and kN Gaussian distributions. Each Gaussian distribution models a landmark of the particle. When the particle collection capacity tends to infinity, the particle filter can approximate the true posterior probability distribution, and its accuracy is close to Best estimate. The initialization of the FastSLAM algorithm: Set the robot pose of each particle to its known starting coordinates, set the map to zero, and then update the particles:

(1) According to the update of the odometer information u_t, a new pose variable is randomly generated for each particle. The distribution of these pose particles is based on the motion model $x_t^{[k]} = P(x_t | x_{t-1}^{[k]}, u_t)$, where the pose at the previous moment $x_{t-1}^{[k]}$ is a part of the particles.

(2) According to the measurement value z_t updates, let the detection road sign number be n, the ideal probability is $\omega_t^{[k]} = N(z_t | x_t^{[k]}, \mu_{t,N}^{[k]}, \Sigma_{t,N}^{[k]})$. Since $\omega_t^{[k]}$ measures the importance of particles based on sensor measurements, it is called importance weight. N represents a normal distribution. Then normalize the importance weights of all particles so that their sum is 1. Finally, re-sampling is performed to keep the particles with reliable observations and update the mean $\mu_{t,N}^{[k]}$ and covariance $\Sigma_{t,N}^{[k]}$ of the particle set. The FastSLAM algorithm has a wide range of applications, it can be applied to non-Gaussian distributed nonlinear random systems, and has a small amount of calculation. It is easy to calculate the importance of the sampling of the motion model and the observation value, making the FastSLAM algorithm one of the easiest algorithms to implement at present.

III. Mobile robot path planning

Path planning is classified into global path planning and local path planning. The global path planning problem can be described as: Finding a collision-free optimal or suboptimal path in the known environment map. In local path planning, the environment map is partially or completely unknown, and the mobile robot uses sensor information to realize its own positioning and dynamic obstacle avoidance. Global path planning is also called offline or static path planning. The main methods include visual method [4], grid method [5] and so on. The global path planning under the known environment map has achieved fruitful results in recent years, so it won't be repeated here. The current research hotspots in the academic circle are mainly concentrated in the field of local path planning. Local path planning is also called online or dynamic path planning.

The artificial potential field method is a method of local path planning. This method assumes that the robot is moving under a virtual force field. Artificial potential field includes attraction field and repulsion field, where the target point generates attractive force on the object and guides the object to move towards it. Obstacles generate repulsive force on objects to avoid collisions with objects. The resultant force experienced by an object at each point on the path is equal to the sum of all repulsive forces and gravity at that point. The attraction field is defined by the following function:

$$U_{att}(q) = \frac{1}{2}\xi\rho^2(q, q_{goal}) \tag{3-1}$$

where ξ is the scale factor, (q, q_{goal}) indicates the distance between the current position and the target position. Gravity is the derivative of the attraction field with respect to distance:

$$F_{att}(q) = -\nabla U_{att}(q) = \xi(q_{goal} - q) \tag{3-2}$$

The following is the repulsion field function:

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\xi\left(\frac{1}{\rho(q, q_{goal})} - \frac{1}{\rho_0}\right)^2, & \text{if } \rho(q, q_{goal}) \leq \rho_0 \\ 0 & \text{,if } \rho(q, q_{goal}) > \rho_0 \end{cases} \tag{3-3}$$

where ξ is the scale factor, indicates the distance between the current state of the object and the target. ρ_0 represents the influence radius of the obstacle. It can be seen from the formula of the repulsion field that when the distance between the robot and the obstacle exceeds a certain threshold, the obstacle has no repulsion effect on the object. The repulsion is the gradient of the repulsion field:

$$F_{rep}(q) = \begin{cases} \xi\left(\frac{1}{\rho(q, q_{goal})} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(q, q_{goal})}\nabla\rho(q, q_{goal}), & \text{if } \rho(q, q_{goal}) \leq \rho_0 \\ 0 & \text{,if } \rho(q, q_{goal}) > \rho_0 \end{cases} \tag{3-4}$$

However, the artificial potential field method still has some problems:

(1) When the object is far away from the target point, the attractive force will become extremely large, and the relatively small repulsion force may even be negligible, and obstacles may be encountered on the path of the object.

(2) When there are obstacles near the target point, the repulsive force will be very large, and the attractive force will be relatively small, so it is difficult for the object to reach the target point.

(3) At a certain point, the attractive force and the repulsive force are exactly equal, and the direction is reversed, the object will easily fall into the local optimal solution or oscillate

In response to problem (1), this paper has made certain improvements to the potential field function and achieved good results. When the distance between the robot and the target point exceeds a certain threshold, the potential field function is improved so that the attractive force received by the robot is reduced, so that the problem of excessive attractive force can be effectively avoided. The corresponding gravity function is changed to:

$$U_{att}(q)=\begin{cases}\frac{1}{2}\xi\rho^2\left(q,q_{goal}\right) & ,\text{if } \rho(q,q_{goal})<d \\ d\rho\left(q,q_{goal}\right)-\frac{1}{2}\xi d^2 & ,\text{if } \rho(q,q_{goal})\geq d\end{cases} \quad (3\text{-}5)$$

In the formula, d is the set threshold. When the distance between the robot and the target point is greater than d, the attraction function will produce a certain correction effect to reduce the attraction effect.

IV. SLAM and path planning under ROS framework

The ROS framework adopts a distributed structure design, and different nodes communicate through a message mechanism. For example: in a mobile robot, data collection is controlled by a laser scanning node and an odometer node, the SLAM algorithm used to process sensor data runs on another node, and the data collection node and the SLAM node implement data exchange by publishing and subscribing to topics. This structure design ensures the real-time data update and data association synchronization of the robot. At the ROS computing graph level, each node does not interfere with each other, even if one node crashes, it will not affect the operation of other nodes.

This paper is based on the FastSLAM algorithm of particle filter to complete the robot real-time positioning and map construction, and make ROS function package by writing and improving the code. The robot base in ROS is represented by the node base_link). Update the robot's position estimation and speed estimation and its covariance matrix through the odom message issued by the odometer, update the time information and laser scanner information related to the data through the sensor_msgs message issued by the sensor, and calculate the cost according to the artificial potential field algorithm introduced above. The shortest path of the map realizes partial dynamic obstacle avoidance.

V. Simulation experiment based on ROS

A. *Intelligent car model and simulation platform construction*

Gazebo is a 3D physics simulation platform with a powerful physics engine, high-quality graphics rendering, diversified programming and graphics interfaces, which can meet the needs of mobile robot navigation simulation. The simulation experiment in this article is carried out on the Gazebo platform.

The ROS operating system provides the function of building simulation models of mobile intelligent robots. URDF (Unified Robot Description Format) is the description format of the robot model in ROS, which includes the description of the appearance, physical properties and joint types of the robot rigid body. At the same time, ROS also provides a C++ interpreter for URDF files, which can parse robot models described in XML format in URDF files. The robot model created in this paper is a three-layer cylindrical robot equipped with a lidar as a sensor to collect environmental model data for mapping work.

To create a complete robot model, you first need to use the URDF format to describe the physical structure of the robot chassis, including: 1 robot base plate, 2 motors, 2 drive wheels and 2 universal wheels, a total of 7 links (describe a certain robot A rigid structure), correspondingly, 6 joints are used to connect each link system together. Then, add <inertial> and <collision> tags to the URDF to add inertia and collision attributes to the chassis model.

After the base plate is completed, the URDF file is converted into an xacro file to realize the reuse of the code, so as to copy the code of the base plate and complete the physical model of the robot. The final result is shown in Fig.1.

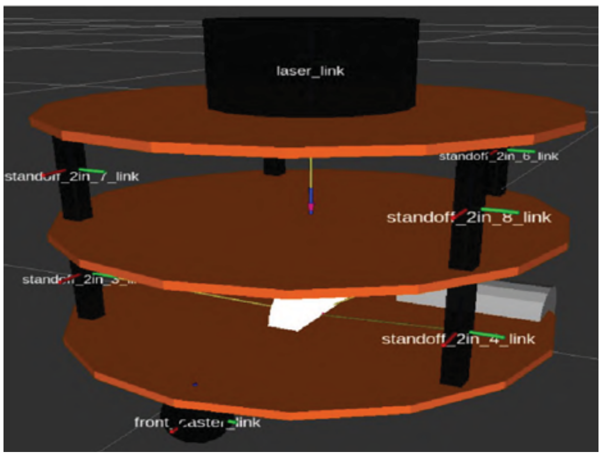


Figure 1 Simulation robot model

After the physical model of the robot is established, it needs to be able to make the robot have power and gazebo attributes. The robot model constructed in this paper is a differential drive robot, which can complete forward, turn, and reverse actions by adjusting the speed ratio of the two driving wheels. In order to use the ROS controller to drive the robot, <transmission> element should be added to the model, and bind the transmission device to the joint. In addition, the Gazebo plug-in is required to complete the simulation output of the sensor and control the motor. Among them, the differential control plug-in libgazebo_ROS_diff_drive.so can achieve the above functions. Declare the plug-in in the corresponding model description file mrobot_body.urdf.xacro, and the robot model can realize the differential speed control function.

B. *ROS configuration*

To configure the function package of SLAM and path planning under ROS, three aspects need to be configured and improved [6-7]: configure the robot through the tf coordinate transformation function package, publish odometer information through ROS, and publish sensor data flow through ROS.

The tf coordinate transformation function package is used to publish the coordinate transformation of the robot, which defines the offset between different coordinate systems. For example: a simple mobile robot can define 2 coordinate systems, base_link is used to represent the base coordinate system, base_laser is used to represent the laser sensor coordinate system, and the data collected by the robot is based on the center of the laser sensor, which is used to realize the obstacle avoidance function of the robot. However, the base coordinate system is used to control the robot's forward direction and rotation posture, so it is necessary to construct a transformation matrix between base_link and base_laser.

The relationship between base_link and base_laser is:

$$T_{\text{base\_link-base\_laser}}=T_{\text{base\_link-world}}\times T_{\text{world-base\_laser}} \quad (5\text{-}1)$$

Firstly, create tf_publisher and tf_subscriber, that is, the publisher and listener of tf transformation. The transformation of base_link relative to the world coordinate system is created in the publisher, and this transformation is broadcast through the sendTransform interface. After the transformation is broadcast, the listener can listen to the transformation information through the waitForTransform interface and the lookupTransform interface. Finally, enter catkin_make in the Linux terminal to compile and run, and the system can obtain the transformation between the base coordinate system and the lidar coordinate system. Publish nav_msgs/odometry odometry information through ROS, which stores the position and velocity estimation of the robot in free space. The sensor_msgs sensor data stream is released through ROS. The message stores the laser scanner information to realize the robot's dynamic obstacle avoidance. The stamp field in the message stores the time information related to the data.

Write the code one by one according to the above steps, use the discrete particle weights to estimate the robot pose PoseArray, and realize global path navigation when the map information is known. Under the condition of unknown environment map, the gmapping function package can construct a map from the data released by the Laser_Scan laser scanner node.

Enter rqt_graph in the Linux terminal, rqt_graph command can call up the ROS system visualization tool. Fig.2 shows the mutual call between nodes when the mobile robot is navigating. The simulation experiment on ROS solves the real-time problem of data association during the robot movement. ROS maintains the synchronization of robot data updates by calling each other, publishing messages and subscribing to topics between different nodes;speed; the code and function package written on ROS can be applied to other robot systems, which is portable and can

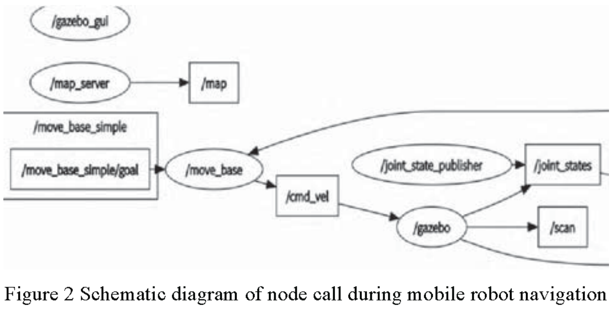


Figure 2 Schematic diagram of node call during mobile robot navigation

improve the efficiency of program development.

C. *SLAM simulation*

The mobile robot in this paper uses Lidar as a SLAM tool. Lidar has many advantages such as high accuracy, fast response, small data volume, and high cost performance. ROS defines a special data structure—LaserScan, it is in the sensor_msg package, which is used to store laser messages. Using the gmapping function package can realize the FastSLAM mapping method based on particle filter. By manipulating the keyboard, the mobile robot can quickly build a map. It mainly uses the particle filter principle for real-time positioning and then uses the grid map mapping method under a fixed path to build an occupied grid map.

Start the Gazebo simulation environment, gmapping node and keyboard control node in the terminal. The keyboard is used in the terminal to control the robot to move in the simulation environment. As the robot continues to move, the map in rviz is constantly updated. After the robot is controlled to explore the location environment completely, the map is basically completed. For the missing part,

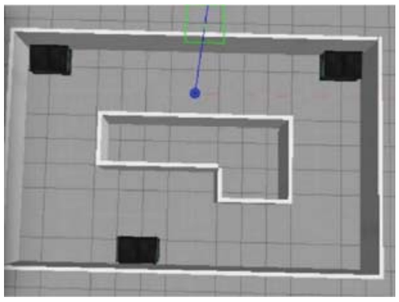


Figure 3 Gazebo simulation environment

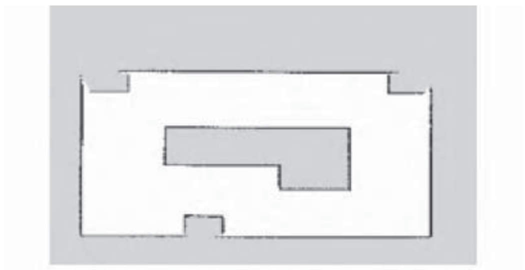


Figure 4 Two-dimensional map created by FastSlam algorithm

continue to control the robot to reciprocate in the corresponding area until the map is completed. Use rosrun map_sever map_save command to save the currently generated map.

Fig.3 is the simulation environment built under the gazebo platform, and Fig.4 is the generated two-dimensional map file.

It can be seen that the effect of using Lidar for SLAM simulation in Gazebo is good, basically restoring the physical environment of the simulation environment.

D. *Smart car navigation simulation*

The key to navigation is positioning, global navigation and local path planning. The ROS framework provides amcl and move_base function packages. The amcl function package provides an adaptive Monte Carlo positioning method. This is a probabilistic statistical method that uses particle filters to track the state of the robot against existing maps. By subscribing to the geometry_msgs/PoseWithCovarianceStamped::amcl_pose topic, the robot's pose estimation with covariance information in the map can be obtained to complete the robot's self-positioning.

The move_base function package contains two planners: a global path planner and a local real-time planner. This paper uses Dijkstra algorithm to plan the global path. The global planner can obtain the motion planning goal by subscribing to the action information of move_base_msgs/MoveBaseActionGoal::move_base::goal. After completing the path planning, output the speed command to the robot chassis by publishing the geometry_msgs/Twist::cmd_vel topic.

This paper uses improved artificial potential field algorithm as the algorithm of local obstacle avoidance. The principle of the artificial potential field method has been introduced above. The main content of algorithm programming is the realization of attraction field and repulsion field. The steps are as follows:

(1) Set up two subscribers to receive the pose data and odometer data of the robot.

(2) Write three-dimensional coordinates and quaternion conversion functions to convert the received pose data and odometer data.

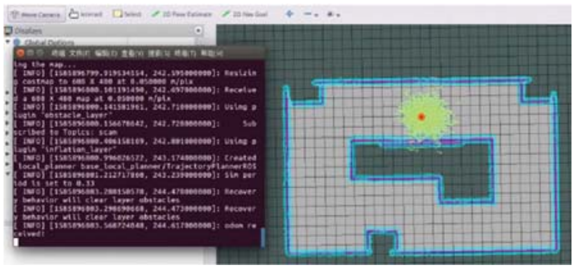
(3) Write the callback function to input the initial position and initial velocity vector of the robot.

(4) Write the attraction function and the repulsive force function, where the function selects the improved artificial potential field function described above.

(5) Write the APF function to calculate the resultant force acting on the robot.

(6) Write the main function to calculate the pose of the robot at the current moment, and calculate the pose of the robot at the next moment as the output of the algorithm through the mechanics formula and the force exerted by the robot and pass it to the controller.

After completing the construction of the potential field, declare the algorithm package and configure the parameters in the corresponding launch file, including the setting of the scale factors of the attarction field and the repulsion field. The results are as follows:



```
ing the map...
[ INFO] [1585096799.919534154, 242.195000000]: Resizin
g costmap to 608 X 480 at 0.050000 m/pix
[ INFO] [1585096800.101191490, 242.497000000]: Receive
d a 608 X 480 map at 0.050000 m/pix
[ INFO] [1585096800.141581961, 242.710000000]: Using p
lugin "obstacle_layer"
[ INFO] [1585096800.156678042, 242.720000000]:     Sub
scribed to Topics: scan
[ INFO] [1585096800.406158109, 242.881000000]: Using p
lugin "inflation_layer"
[ INFO] [1585096800.996026522, 243.174000000]: Created
 local_planner base_local_planner/TrajectoryPlannerROS
[ INFO] [1585096801.212717800, 243.239000000]: Sim per
iod is set to 0.33
[ INFO] [1585096803.288150578, 244.470000000]: Recover
y behavior will clear layer obstacles
[ INFO] [1585096803.298690668, 244.473000000]: Recover
y behavior will clear layer obstacles
[ INFO] [1585096803.568724848, 244.617000000]: odom re
ceived!
```

Figure 5 The overall effect when the navigation algorithm is running

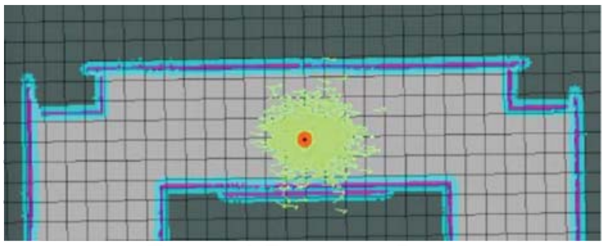


Figure 6 State diagram of mobile robot

The mobile robot in Fig. 5 and Fig.6 is moving towards the designated target point on the map that has been built. The dark green arrow is the global path for navigation, and the light green arrows are the join forces force of the repulsive force and attractive force during the robot's travel. If the original artificial potential field function is used, when the target point is set far away, the robot may collide with obstacles at the corner. After the potential field function is improved, the robot can reach the target location smoothly, so that the simulation result is good.

VI. CONCLUSION

This paper studies the SLAM problem and path planning problem of mobile robots, and establishes a mathematical model of the SLAM problem. FastSLAM algorithm based on particle filter is adopted. In terms of path planning, the problem of local path planning based on sensor information was analyzed. The dynamic obstacle avoidance of mobile robots was realized through the artificial potential field method. Finally, the expectations were obtained.

REFERENCES

[1] Smith R, Self M, Cheeseman P. Estimating Uncertain Spatial Relationships in Robotics[J]. Machine Intelligence & Pattern Recognition, 1988, 5(5):435-461.

[2] Arulampalam M S, Maskell S, Gordon N, et al. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking[J]. IEEE Transactions on Signal Processing, 2002, 50(2):174-188.

[3] A Monte Carlo Approach to Nonnormal and Nonlinear State-Space Modeling. 1992, 87(418):493-500.

[4] Yang Huaiqing, Xiao Xinggui, Yao Dong. A Robot Global Path Planning Algorithm Based on Visualization Method[J]. Journal of Shenyang University of Technology, 2009, 31 (2): 225-229.

[5] Wang Xingce, Zhang Rubo, Gu Guochang. Robot Global Path Planning Based on Potential Field Grid Method[J]. Journal of Harbin Engineering University, 2003, 24 (2): 170-174.