World Scientific Research Journal Volume 6 Issue 11, 2020
ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

Modified RRT*-Connect Guided By APF-based Multiple Weight Reattribution

Rui Chen[1], Bowen Tan[2], Yukang Liu[3], Yihao Zhong[4], Haoran Peng[5], Jiadong Li[6]

[1]College of Control Science and Engineering, Zhejiang University, ZheJiang, HangZhou 310058, China.

[2]Woodward Academy, Atlanta, GA 30337, U.S.

[3]College of ENGINEERING & COMPUTER SCIENCE, Australian National University, Canberra ACT 2601, Australia.

[4]Duke Kunshan University, Suzhou, Jiangsu 215316, China.

[5]University of Vermont, Chengdu, Sichuan 610065, China.

[6]Dulwich International High School, Suzhou 215021, China.

Abstract

The project is intended to improve the RRT* algorithm by combining the artificial potential field and Multiplicative Weights methods. First of all, for a map composed of a random 0-1 matrix, each obstacle point has a potential field, and a virtual force is exerted by each point on the vehicle. All the forces are added together to calculate a joint force, which represents the recommended direction of RRT expansion. Then, according to this recommended direction, we readjust the probability of each sampling node extending in different directions to form a new heuristic algorithm for graph-search procedure based on the RRT*-connect method.

Keywords

APF, RRT (*)-connect, Matrix Feature extraction, Deep Learning, Multiple Weights Reattribution, Normal distribution.

1. INTRODUCTION

1.1. Motion planning

The basic task of robot motion planning can be described as the movement from the starting position to the target position. This task usually involves a fundamental issue of how to avoid obstacles in the given space (geometric path planning). Sampling-based motion planning algorithm is an important method for solving the problem, and its core idea is: perform collision detection on a single configuration of the robot, establish a database of collision-free configurations, and then sample different configurations to generate collision-free paths. Typical sampling planning methods include two types: multiple-query method and single-query planning. The former constructs a road map and then builds a complete undirected graph through sampling and collision detection to obtain the complete connection properties of the configuration space. After these procedures, it searches through the graph to get a feasible path. The latter builds a roadmap locally from a specific initial configuration, extends the tree data structure in the configuration space, and finally connects them. Since the distance we extend each time is fixed, there is no guarantee that the last extension will just reach the end position, and it is more likely to jump back and forth around the end point. Therefore, we set a threshold:

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

if the distance between the new point and the end point of this extension is less than this threshold, we think that the planning has been successful.

1.2. Rapidly exploring Random Tree

The rapidly exploring random tree (RRT) is known as a sampling-based motion planning algorithm with single-query planning that can search through nonconvex, high-dimensional spaces by randomly developing a space-filling tree [1]

The tree begins at a starting configuration (for a two-dimensional graph, it is a point), continuously extends the tree data, and eventually connects to the target point. After the starting point is determined, the tree continues to grow toward the target point. However, it should be noted that due to the existence of random obstacles, if the tree extends toward the target point directly, the tree may fail because of "obstacle crashing". Therefore, the RRT adopts a random sampling method: each time, the growth direction is selected with a certain probability so that it will extend toward the target point. There is also a certain probability that it will randomly choose a direction in the map to extend a distance. The simplest one is that it will extend toward the target point with 50 percent probability. There is also a 50 percent probability that it will randomly choose a direction to extend a distance. This process is a random sampling of the search space. Since there is a certain randomness in each growth, many branches will gradually appear in the RRT tree. Then, the algorithm will directly select the point closest to each sampling point in the RRT tree and extend it. If no collision occurs during this extension and the distance between the new point and all existing points is greater than a certain threshold (to prevent growth to the position that the RRT has already explored), the method will add this new point to the RRT tree.

RRT tree development can be biased by expending the probability of the examination of states from a particular range. This is accomplished by introducing a small probability of sampling the goal to the state sampling procedure. Most practical implementations of RRTs make use of this to guide the search towards the planning problem goals. The higher this probability, the more greedily the tree grows towards the goal.

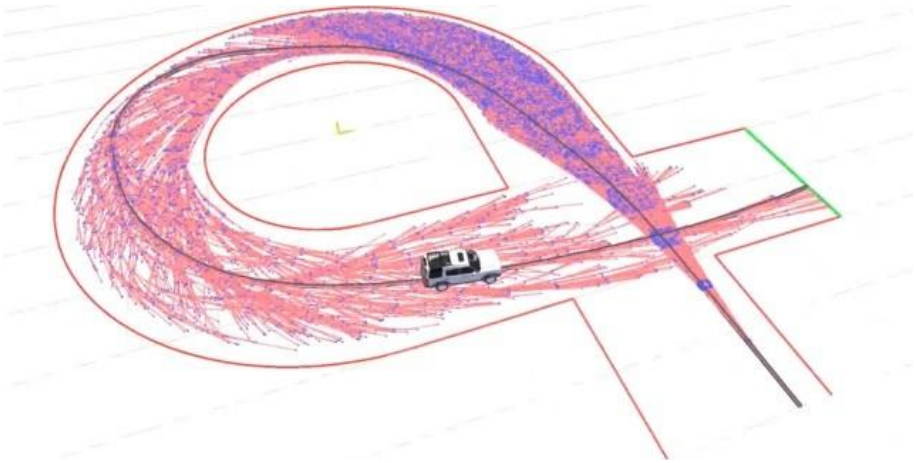

Figure 1. RRT-planning

1.3. RRT* and RRT-Connect

RRT* inherits all the properties of RRT and works similar to RRT. However, it introduces two promising features: near neighbor search and rewiring tree operations. Near neighbor operations finds the best parent node for the new node before its insertion in tree. Rewiring operation rebuilds the tree within this radius of area to maintain the tree with minimal cost between tree connections [2]. The RRT-Connect planner is a variation of RRT. It is designed

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

specifically for path planning problems that involve no differential constraints. It has two unique properties: it grows two trees from both the start and destination until they meet instead of one tree, and it grows the trees towards each other (rather than towards random configurations). It is the improved version of RRT for faster convergence.

Each sampling has a certain probability to move towards any direction or toward the end. This probability will obviously affect the search effect. The most direct impression is that the greater the probability of random sampling, the more branches of the RRT tree. We developed a practical implementation of RRTs by using this to guide the search towards the targets. We also applied the multiplicative weights method, which updates the weight of optimal probability based on the result of artificial potential field, and, after certain update, it provides more effective direction.

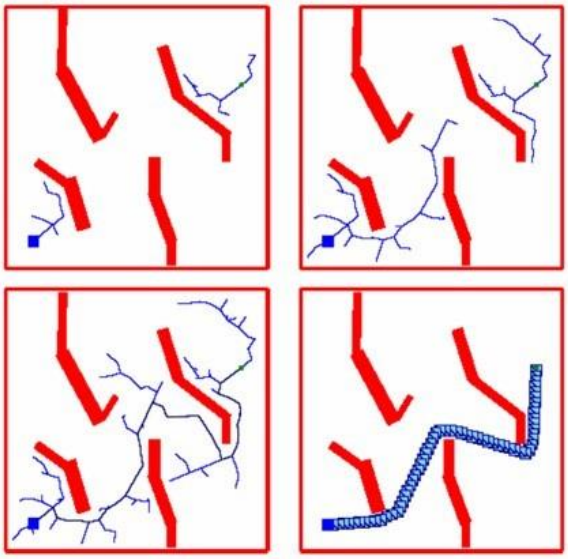

Figure 2. RRT-connect

1.4. Application

The improved RRT algorithm has an improved heuristic algorithm, which will help to make the "random" process of RRT more biased. In addition, this will make the RRT star and the RRT Connect have more potential use.

First of all, the RRT can help to minimize the problem of ocean current when the robot functions underwater. Moreover, the algorithm can also consider the complex terrain environment when finding the route for the vehicle. The result of the algorithm can be reliable to consider when choosing the shortest route for the vehicle underwater.

Secondly, the algorithm can also be used for drones to determine the appropriate route to go. However, the current way of RRT takes too much time to find the correct path and will not be suitable for the drones. The connect RRT or the RRT star, on the other hand, can compute the path in a short period and is well suited for finding the way during a complex environment (Such as the forest). Furthermore, the algorithm will also help for the drones to change the path during the flight in case of emergencies, which credits to the ability of the RRT* to find the path in a relatively short time.

In addition, some autonomous vehicles can also use the RRT as a way to find the route. In the complex real-world situation, finding a route for the vehicles in a short period of time is vital for the autonomous vehicles, and the RRT* or the connect RRT is an efficient way to get the correct route in a short period of time. First, the car needs to take samples of the real-world

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

situation and generate a graph for the RRT* or the connect RRT to perform. Then, after the graph has been generated, the algorithm could be applied and find the path for the vehicles.

Finally, one of the potentials uses of the RRT is to determine the game level of different games. The idea is to provide a visual representation of the game level for the developers of the games. The game graph is computed by the algorithm and the resulted states and transactions are processed to show the game level. The advantage of using improved RRT is clear as the process of clustering will not be needed. At last, the average time taken is calculated and the game level is assigned with the game.

2. ARTIFICIAL POTENTIAL FIELD

2.1. Brief Overview

Artificial potential field (APF) method is a classic path planning algorithm. The algorithm treats targets and obstacles as objects that have gravitation and repulsion to the robot, respectively, and the robot moves along the join force of gravitation and repulsion. For example, it transforms the configuration space to a potential field plane and represents the robot (the current configuration) as a point in space. If the starting point and obstacle of the robot are positively charged, the end point is negatively charged, and the robot is positively charged. The robot will move along a certain path to the end point under the action of electric field force and avoid positively charged obstacles.

2.2. Why and How We Apply APF Into the Project

Because of the largely random nature and the lack of process on the environment of RRT algorithm, the efficiency of the algorithm will be negatively affected by its blindness of its surroundings. To solve this problem, the Artificial Potential Field algorithm, an algorithm which processes and stores information about surrounding obstacles and the target, is implemented to improve the efficiency of the algorithm. The APF algorithm does not guarantee to produce the shortest path to the target, but it can produce a smoothest and safest path [3].

By regarding the moving vehicle object as a point, the Artificial Potential Field algorithm simplifies the calculation of the fictional combined potential the vehicle experiences: the fictional repulsive potentials exerted by obstacles and attractive potential exerted by the target are to be combined to generate the resultant combined potential. With the potential field function, the vehicle will move in the direction of descending potential, thus avoiding encountering the obstacles.

In the research paper by Oussama Khatib, the Artificial Potential Field algorithm was implemented to facilitate the moving robot to avoid obstacles [4]. The fictional potential energy function can be represented by adding the attractive potential and gravitational potential experienced by the robot:

$$U_{art}(x) = U_{goal}(x) + U_{obs}(x) \tag{1}$$

$$F_{art} = F_{x_d} + F_{obs} \tag{2}$$

In the equation (1), the attractive potential is the sum of the potential from the goal and the potential from obstacles. In the equation (2), the attractive force is the sum of the force from the robot to the goal and the force by obstacles, and $x_d$is the goal position.

$$U_{x_d}(q) = 0.5k_{att}(q - q_d)^2 \tag{3}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

$$U_o(q) = \begin{cases} 0.5k_{rep}(\frac{1}{\rho(q)} - \frac{1}{\rho_o}) & \rho(q) \le \rho_o \\ 0 & \rho(q) > \rho_o \end{cases} \quad (4)$$

In the equation (3), $k_{att}$is the attractive force gain coefficient and x and$x_d$represents the position of the robot and the goal. In the equation (4), k_{rep}represents the repulsive force gain coefficient and $\rho(x)$represents the Euclidean distance from the robot to the given single obstacle and $\rho_o$represents the threshold for maximum distance of influence of the repulsive force.

The fictional forces the robot or vehicle experiences from the goal and obstacles are negative gradients of the attractive and repulsive potential functions respectively. Thus, by calculating the negative gradients of the attractive and repulsive potential functions, we obtain:

$$F_{x_d}(q) = -grad[U_{x_d}(x)] = -k_{att}(q - q_d) \quad (5)$$

$$F_o(q) = -grad[U_o(q)] = \begin{cases} k_{rep}(\frac{1}{\rho(q)} - \frac{1}{\rho_o})(\frac{1}{\rho^2(q)})(\frac{q-q_{obs}}{|q-q_{obs}|}) & \rho(q)\rho(q) \le \rho_o \\ 0 & \rho(q) > \rho_o \end{cases} \quad (6)$$

However, the above equation is only for calculating the repulsive force exerted by a single obstacle. To compute the combined potential, the net repulsive potentials from all obstacles and the attractive potential must be considered and the net repulsive forces and the attractive force must be also added to generate the combined force.

$$U(q) = U_{x_d}(q) + \sum_{i=1}^{n} U_o(q) \quad (7)$$

$$F(q) = F_{x_d}(q) + \sum_{i=1}^{n} F_o(q) \quad (8)$$

However, there are two problems with the Artificial Potential Field algorithm: first, it can be easily trapped in local minimum points; second, the attractive force is proportional to the absolute distance from the robot/vehicle to the goal, leading to the robot encountering obstacles when the repulsive force is negligible compared to the attractive force.

For the first problem, the Artificial Potential Field algorithm is implemented in the RRT algorithm with multiplicative weights. Because of the randomness of the RRT algorithm and the determining multiplicative weight algorithm, the robot will start relatively randomly after a local minimum point is reached, and the local minimum point will not be converged so easily.

For the second problem, a threshold for the distance of influence of the goal is set to limit the infinitely increasing attractive force.

$$F_{x_d}(q) = \begin{cases} -k_{att}(q - q_d) & (q - q_d) \le d \\ 0 & (q - q_d) > d \end{cases} \quad (9)$$

In this equation, the attractive force is set as a constant number once the absolute distance between the vehicle and the goal is larger than the set threshold, thus effectively limiting the increase of the attractive force and the absolute distance from the robot to the goal increases infinitely.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

Algorithm 1 APF Calculation

```
procedure EXTRACT_OBSTACLE_EDGE
    Traverse the map matrix.
    for row in rows do
        epcc ← Record edge point column coordinates.
    end for
    for col in columns do
        epcr ← Record edge point row coordinates.
    end for
loop:
    list_of_obstacles_pos ← combine(epcc,epcr)
    goto loop.
    return list_of_obstacles_pos
end procedure
procedure CALCULATE_COMBINEDFORCE
    stringlen←length of string
    i←patlen
loop:
    for obstacle in list_of_obstacles_pos do
        dist ← distance(vehicle, obstacle)
        repulsion ← 0
        if dist != 0 and dist < maxdist_of_repulsion then
            repulsion ← repulsion + repulsionForce(vehicle, obstacle)
        end if
        attraction ← attractionForce(vehicle, obstacle)
        Xforce ← XcombinedForce(attraction, repulsion)
        Yforce ← YcombinedForce(attraction, repulsion)
    end for
    goto loop.
    return (Xforce, Yforce)
end procedure
```

Figure 3. Pseudocode of APF generation

3. MAP GENERATION AND FEATURE EXTRACTION

To test the reliability of our idea, we decided to make matrix with a simple structure (0-1) to make our idea backed up with evidence. For testing the performance of our idea easier, we generate our matrix randomly with a specific proportion (70%) and then test the connection. The merits of testing the connections of the matrix are that testing could provide a matrix where there always could be a solution between each pair of available points. Hence, when we test our idea, we can select multiple pairs of points in the same matrix for many times without consideration of failing to find a way. That enlarges our samples significantly. Besides, we could record the best solution of the desired pair of the points so that we could regard the results as the expected ones.

For algorithm of testing matrix full connected, in the testing phrase, we used recurrence to crawl all possible way in the all possible directions. The time complexity was current, and m represents the possible directions of every points (4 in 2-d, 8 in 3-d) and n means all the available points. Results show that it works well in the small sized matrix.

However, when we apply our idea in practical problem, the map will become more complex in size, dimension, etc. Besides, we cannot only use 0-1 matrix to simulate all the maps. In that way, the time cost of the recurrence algorithm will rise exponentially. Hence, we need to find a

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

reliable and efficient planning optimization algorithm to replace it. In this project we used Ant Colony Optimization, which derives from the ant communication system. [5]

In an ant group, ants can find the best way to food in various environments because they can generate pheromone. Normally speaking, the more concentration of pheromone one path has, the more probability it will be chosen. Then the ant will leave the pheromone in that pathway as well, increasing the pheromone concentration in that path. Using this mechanism, all the pathways could converge the best solution to food in the end. Hence, we could use that mechanism to solve the problem.

Here we are given the more general problem in a complex graph with different path length. We have V as the set of points, E as the set of edges. Then we regard η as a heuristic value, which relies on the pathways from any pair of points. Besides, we set α as the pheromone, which is positively correlated with the pathways as well. The initial value of α we set is$\frac{1}{|V|Mean}$, where $|v|$ represents the number of available points, and mean represents the average length of E. At the start of the algorithm we set up m agents (i.e. random selector) and force them to generate the path to the destination and leave the pheromone along the path. Then when agent need to go to the next available points, we need to make a specific probability distribution to influence their choice, which is:

$$P_{rs}^{k} = \begin{cases} \frac{\alpha_{rs}^{\beta}\eta_{rs}^{\varepsilon}}{\sum \alpha_{rz}^{\beta}\eta_{rz}^{\varepsilon}} & s\ belongs\ \ to\ \ \ J_k(r) \\ 0 & otherwise \end{cases} \tag{10}$$

Where p_{rs}^{k} is the probability of the ant k in point r choosing the points, $J_k(r)$ presents the set of points where ant k in the current location r can choose. Band ε is the weights of pheromone and heuristic value respectively.

Procedure 1 Create maze

Input: Assigned dimensions of the maze and specific size of each dimension. To simply the procedure, we set our dimension as 2.

Output: the desired maze which contains has one or multiple paths starting from entrance to the exit

```
while True do
    for row in rows do
        for column in columns do
            set every cell as 1(hinder)
            set element in the top left and bottom right as entrance and exit
respectively.
            if Not entrance nor exit then
                random assign 0,1 to these elements with specific distribution
            end if
        end for
    end for
    start from the entrance;
    if if find a way in the procedure then
        return the maze
    end if
end while
```

Figure 4. Pseudocode of maze creation

From all the math models, we could make the paths for the ants, when ants finish the tours (successful or not). We could update the pheromone as follows:

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

$$\alpha_{rs} \leftarrow (1-p)\alpha_{rs} \; for \; all \; (r,s) \tag{11}$$

$$\alpha_{rs} \leftarrow \alpha_{rs} + sum(\alpha_{rs}) \; if \; (r,s) \; is \; the \; best \; solution \tag{12}$$

When an iteration ends, all the pheromones will decrease with a specific proportion. And we rise the pheromone in the point which is in the current best way. For several iterations, it will return a general best solution.

Procedure 2 Find a way

```
Input: Unchecked Maze and its current position
Output: Boolean to show whether there is an effective way from the entrance
  to the exit or not
  if current position is the exit then
     return True
  end if
  for direction in all possible directions do
     if new direction is available then
        algorithm with the maze but new position.
     end if
  end for
  return False
```

Figure 5. Pseudocode of standard maze generation

After going through these, we could generate a stable and testable map dataset. Further, we could make a supervised neural network model to pick up the general features of a map without knowing the details through iterating.

4. OVERALL PROCESS OF THE ALGORITHM

4.1. How We Come up with the Algorithm

In this section, we will show the specific implementation of this project. The following part of report will be presented and organized in the form of a series of problems and how we use the advantage of other methods to solve them. In fact, both the RRT and APF themselves are not the most advanced and commonly used methods in their respective fields, because they have some obvious defects, but some of these shortcomings can be compensated by the advantages of another method

For example:

The disadvantage of artificial potential field is that it runs slowly and it is easy to trap into a non-optimal "local maximum" just like some gradient descent algorithm (In reality, this is reflected in that the path will cycle in one place during planning)

Solution:

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

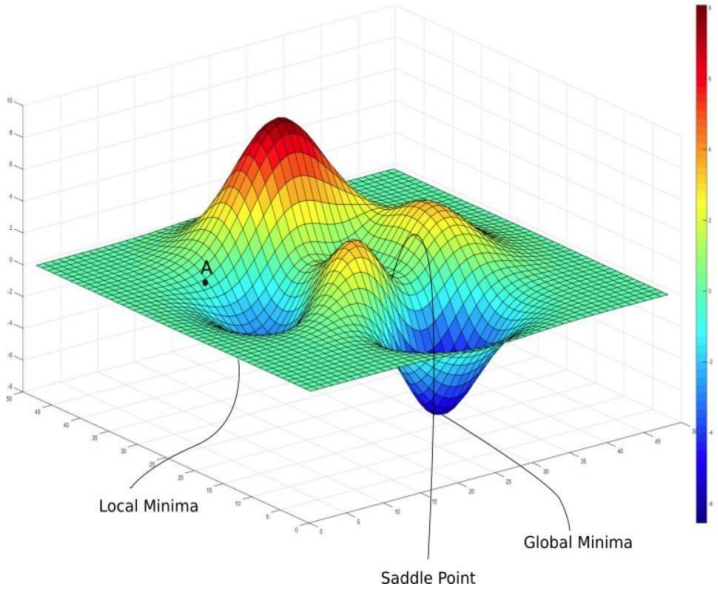

Figure 6. Local-maximum/minimum

We can use the random sampling of RRT to help the APF, which is trapped in local minimum, to escape from the dilemma. Just like the simulated annealing algorithm and other gradient descent algorithm falling into the local maximum and minimum value, once the judgment mechanism is activated, it will jump out and continue to explore.

Another example is:

The biggest defect of the RRT(*) itself is its uncertainty: it cannot guarantee to find the optimized path within specified iterations.

And the solution:

The guidance of artificial potential field can discard some of the wrong attempts to overcome this defect.

So, we use the APF as the main body of the heuristic guiding mechanism, but When the two are combined, we will find many interesting problems.

4.2. Mechanism of Reattribution

The macro problem is also obvious:

what should we do after we get this resultant force at some point in the RRT expansion process?

Obviously, we can extend RRT directly to a given location, but in this way, RRT will lose its meaning and is not a random algorithm. So, the weather-forecasting example of Multiple Weights Reattribution come into my mind:

We intend to increase the weight of this resultant force direction and reduce the weight of other directions, especially those opposite directions. And in RRT's case, the "weight" is just the probability of generating a new node and adding it to the RRT. But this is just a vague idea so far.

But when we started writing code, we found it hard to put this idea into practice, because first of all

How should we express this direction in the code?

If we simply calculate, then it involves trigonometric function and anti-trigonometric function and multiple four arithmetic operations, which is very time-consuming.

And we know that in RRT expansion process, every time we want to create and add a new point, we first randomly generate a point on the graph(this "new node" is not the new node

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

which is going to be added into the RRT, it only serves as a tool to realize "random direction" in the most time-saving way), and use the line between this point and another point in existing RRT to indicate the direction. And

In the traditional RRT, once a new random point is set, there will be 50% probability to expand toward the new random point and 50% toward the destination point.

So here comes the next question:

What should we do if we want to reattribute the probability and also preserve the precious time-saving characteristic?

Our solution is:

We could get the idea from the example of Modifying the weight of experts according to the weather forecast results. We set up 100 angles to represent 100 experts. The resultant force calculated before is equivalent to the result of the weather forecast.

The problem now is:

How to make a mechanism to set the values of these angles (like experts)?

We knew that

A force can be represented by its size(double) and direction(angle), in this case, We intuitively felt that these angles that were going to be assigned should be as close to the angle of the joint force as possible. That is, the closer to the angle value of the combined force ,the higher the probability of being sampled should be; the further to the angle value of the combined force ,the lower the probability of being sampled should be. And for the value of the joint force generated by the obstacle's artificial total potential field, how should we deal with it? In fact, that's very simple, the larger the value of the force, then the sampling probability should be more concentrated around this, the opposite should be more scattered.

Therefore, we set the mechanism as follows:

If you are similar with normal distribution or Gaussian distribution, you will find that ND could be used to complete the assignment of probability.

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{13}$$

This is the standard expression of normal distribution. σ is the scale parameter, which is the variance of normal distribution. μ is the position parameter. The X range in this is ± PI. μ is the angle of resultant force. We made σ inversely proportional to the size of the resultant force.

And In coding part We originally set σ as π/4.

And the diagram is showed as follows:

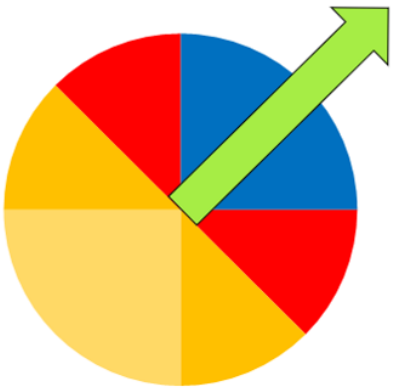

Figure 7. Probability Redistribution

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

The larger the variance of normal distribution σ is, the more dispersed the value of the random variables are, the lower or wider the image will be. We also know that σ represents the degree of dispersion of the values taken by random variables. The smaller the value, the higher or narrower the image, the higher or narrower the value of the random variable. The higher the value, the more dispersed the value of the random variable, the lower or wider the image.

Thus, we let the value of the combined force to be inversely proportional to the variance of normal distribution σ.

Here we have a second coefficient to train: the coefficient between σ and inverse proportion of the value of force.

5. RESULTS AND ANALYSIS

5.1. Visualization and Analysis of APF and RRT*-Connect.

We used MATLAB's mesh function to display the 3D effect of the whole artificial potential field. For clarity, we showed the fields produced by attraction and the fields generated by repulsion separately and their combination.

Then we saw the effect of RRT, which didn't look much different from that of RRT*.

Next we combined this path with the diagram, and we could clearly observe the obstacle avoidance effect of this method

5.2. Visualization and analysis in ROS's Wheeled Robot Simulation

We visualize the actions of the 100 particles (correspond to 100 angle groups (in ND)), But because my communication with ROS is not complete, the visualization of these particles has some delay, shows the redistribution of weights in the normal distribution that We mentioned earlier. Also, we visualizes the application in wheeled robots, the arrows in the video represent the RRT expansion path in in the current state (Attention: The visualization takes place after the RRT is completed, and the car will follow the path which RRT constructed. In the process, the arrow will show the direction of combined force in the nearest node of the current location of the vehicle. The different colors indicate the RRT* and Modified RRT we have created).

5.3. Visualization and Analysis in ROS's Robot Arm Trajectory Planning

Then we visualize the application in robotic arms. The DEMO4 require the end of robotic arm to touch the red ball and blue ball back and forth recursively, in the process the robotic arm should avoid to touch the obstacle (I use cylinder to represent it). In DEMO 5, We just Set the starting point and the destination point, and use modified RRT to plan the path and move the robotic arm! We can see the result is very satisfying.

5.4. Surprising Results

So far, we might not feel that the result of this algorithm is perfect. But In fact, the pictures below actually came from two different forms: the picture on the left was the result of the overall algorithm, but the picture on the right was another RRT that we run in parallel in the same program. The difference between this RRT and the original one is that we canceled the collision detection link; that is to say, this RRT achieves natural collision free without collision detection at all. This was the most surprising result of this experiment.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

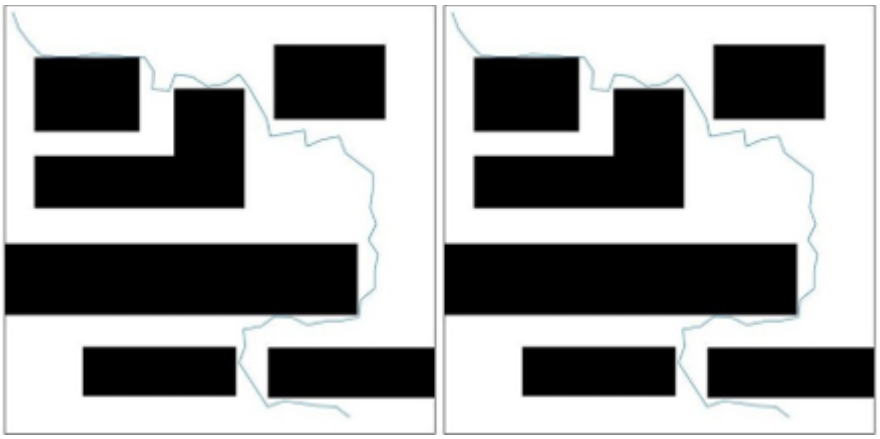

Figure 8. Surprising results

6. IMPROVEMENTS

6.1. Strassen Fast Matrix Multiplication

We knew that time consumption is vital in RRT path planning, so we tried many approaches to do acceleration work. One significant way is to use Strassen Fast Matrix multiplication to replace the normal one. It really worked in python, but the result of time cost tended to be higher than the original matrix multiplication in MATLAB; the result in MATLAB was opposite to what the theory revealed, and after reading the official file of MATLAB, we knew that the Matrix multiplication of MATLAB had already been optimized! Simply applying Strassen Fast Matrix Multiplication only increased the space complexity of it because Strassen Fast Matrix Multiplication needed nested calls which require large contemporary storage.

6.2. Detail Correction

What we have just solved are macro problems. Now let's focus on some details

What if an end point is close to an obstacle?

If we do it according to the traditional algorithm, the repulsion force tends to infinity when approaching the end point, then RRT is very difficult to get close to the end point, which is contrary to our goal.

So, we add an upper bound to the repulsion function. When the repulsion force is greater than this value, we assume that the repulsion force is always this value.

6.3. Deep Learning of Parameters

The next part is to use deep learning to improve the parameters

We had a lot of parameters so far,

Such as

The coefficient between the repulsive force and inverse ratio of the square of distance.

The coefficient between the attraction force and inverse ratio of the square of distance, and the number of the power of distance.

How exactly should we effectively break away from the local maximum value mentioned before, which meant that for those directions that are almost opposite to the direction of the resultant force, we still had to give them a certain probability value. We divided the normal distribution into four regions (corresponding to four 90 degrees region in ± 180 degrees). In this part, we set the initial value of σ as 45 degree. And the coefficient k in

$$\sigma = \frac{k}{|F|} \tag{14}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202011_6(11).0039

While |F| is the value of the combined force.

We knew that for different maps with different characteristics, we should adjust the parameters, so we intended to use deep learning to solve this problem.

We have now built a neural network framework based on CTC + CNN frameworks. After the extraction of map features is complete, we can input the generated feature vector into the neural network for training. In the future, for the existing maps, we can input the trained model Each parameter value is brought into the algorithm.

7. CONCLUSION

The traditional original RRT algorithm does not guarantee the solution in theory. Let alone the optimal solution, compared with the classical ASTAR algorithm, RRT cannot reasonably discard some of the poor historical paths of the past according to the dynamic change of cost and RRT after each new node to update his parent node, to some extent to improve the problem. Because of the complete randomness of the traditional RRT, it cannot find the optimal path, even if there is a certain probability to find, cannot complete the task in a predictable time, so we add a certain bias to the random algorithm of RRT, so that it can find the best path while retaining the advantages of the RRT in path exploration.

The results run in MATLAB and Python show that although the above algorithms and solutions do not give the optimal path, they can make the final path better than the traditional RRT algorithm and discard the collision detection process without causing any inefficiency. At the same time, we take the optimization algorithm when calculating artificial potential field, so that the time complexity of artificial potential field and collision detection time complexity in the same order of magnitude, so the total time complexity has no obvious additional consumption.

In summary, this algorithm provides RRT-series algorithms with a set of feasible solutions and ideas. We believe that in the future, after we train the right parameters through machine learning, this program will be able to perform better.

REFERENCES

[1] LaValle, Steven M. (October 1998). "Rapidly-exploring random trees: A new tool for path planning" (PDF). Technical Report. Computer Science Department, Iowa State University (TR 98–11).

[2] Noreen. I, Khan. A, Habib. Z, (October 2016). “A Comparison of RRT, RRT*, and RRT*-Smart Path Planning Algorithms.” IJCSNS International Journal of Computer Science and Network Security, VOL.16 No.10.

[3] J. Sun, J. Tang and S. Lao, "Collision Avoidance for Cooperative UAVs With Optimized Artificial Potential Field Algorithm," in IEEE Access, vol. 5, pp. 18382-18390, 2017, doi: 10.1109/ACCESS.2017.2746752.

[4] O. Khatib, ‘‘Real-time obstacle avoidance for manipulators and mobile robots,’’ Int. J. Robot. Res., vol. 5, no. 1, pp. 90–98, 1986, doi: 10.1177/027836498600500106

[5] Dorigo, M., Birattari, M. and Stu¨tzle, T., 2020. Ant Colony Optimization. [eBook] Universite´ Libre de Bruxelles, BELGIUM, pp.1-12.