Information Science manuscript No.
(will be inserted by the editor)

Boosting path planning with GAN Predictions

Kun He and Fan Min*

Received: date / Accepted: date

Abstract Path planning is a crucial task in robotics. Sampling-based algorithms have garnered significant attention for this purpose. However, traditional sampling-based algorithms suffer from poor initial path quality and low efficiency. In this paper, we present a new learning-based path planning method. First, we utilize a generative adversarial network to learn from previous successful planning experiences Second, use it to restrict the sampling area in a new map. We then employ RRT* to generate a better initial path by performing path planning within this restricted area. We propose a total of 20 maps, each with varying complexities. Out of these, 16 are used for training and testing, while the remaining 4 are used to test the proposed method's generalization ability. We compare our approach with three state-of-the-art methods. The results demonstrate that our method outperforms them in both path quality and convergence speed.

Keywords Sampling-based path planning · Optimal path planning · GANs

1 Introduction

Robot path planning refers to finding a safe and feasible path for the robot in a given map state space, so that the robot can reach the target position from the starting point. According to the nature of the method, common robot path planning algorithms can be divided into the following categories:

Grid-based path planning algorithms: These algorithms usually divide the map into a grid graph, and then use the shortest path search algorithm to find the optimal path. For example Dijkstra's algorithm [1], A* [2], D* algorithms. This

Corresponding author. E-mail: minfan@swpu.edu.cn (Fan Min).

Kun He
School of Computer Science, Southwest Petroleum University, Chengdu 610500, PR China
E-mail: hk20201008@163.com

Fan Min
School of Computer Science, Southwest Petroleum University, Chengdu 610500, PR China
E-mail: minfan@swpu.edu.cn

type of method transforms the path planning problem into a path search problem in the graph, and the efficiency of this type of algorithm is related to the path quality and the resolution of the grid division.

Bionic-based path planning algorithms: These algorithms typically use biological information to explore maps to guide robot path planning. Such as genetic algorithm, particle swarm algorithm, ant colony algorithm. These algorithms are simple to implement, but require multiple iterations to obtain suboptimal paths, and are prone to getting stuck in local minima.

Sampling-based path planning algorithms: These algorithms usually use random sampling to explore the map, use a tree or graph to store path information, and then use the shortest path search to get the final path. For example Rapidly Exploring Random Tree (RRT) [3], Rapidly Exploring Random Tree-star (RRT*) [4], Probabilistic Roadmap (PRM) [5]. This type of algorithm is fast, but it is usually difficult to obtain the optimal path.

Rapidly Exploring Random Tree (RRT) is a popular algorithm in robot path planning. RRT is a very effective path planning algorithm that can cope with various complex environments. And it is easy to extend to high-dimensional space and add dynamic constraints, and more importantly, it guarantees probabilistic completeness. However, the quality of the initial paths obtained by RRT is poor because it is prone to unnecessarily long paths or zigzag paths. To address this issue, researchers have developed several variants of the RRT algorithm, such as RRT*. It is an improvement over the basic RRT algorithm. RRT* adds neighbor search and rerouting to ensure asymptotically optimal path quality. In short, RRT* optimizes the existing paths while exploring the space, thus ensuring asymptotic optimality. But to obtain high-quality paths means more iterations are needed, which means that RRT* cannot achieve a balance between path quality and efficiency.

To alleviate this shortcoming, we propose a path planning method based on Generative Adversarial Networks. Our algorithm speeds up path planning by constraining the search space. First, our model is trained using successful planning experience to obtain a path planner. Subsequently, the restricted sampling area is predicted using the trained planner. Finally, a sampling-based approach is used to sample over the region. The restricted sampling area can avoid RRT* blind search, thus speeding up the sampling process and obtaining better initial paths.

Our contributions are summarized as follows:

1) A new path planning method is proposed.
2) A new path planner is proposed.
3) A new path planning dataset is proposed and the feasibility of our scheme is verified.

2 Related work

In the past few years, there have been many research works dedicated to improving the performance of RRT algorithms. RRT-connect proposes a new way to iterate over tree nodes. It uses the start and end points as the root nodes of the two trees, respectively, and alternately iteratively explores the state space until the two trees meet. The efficiency of RRT-connect exploration of state space has been significantly improved, but the path it generates still needs to be optimized.

Informed-RRT* uses the same iteration and optimization strategy as RRT*. However, IRRT* implements heuristic search for ellipses, which makes IRRT* converge faster.

In terms of improving the RRT algorithm, there are a series of research works dedicated to improving the path quality.

Using deep learning models to improve the performance of RRT is a promising direction.

3 Preliminaries

In this section, we briefly introduce the related concepts of path planning, the Markov clustering algorithm, and RRT*.

3.1 Path Planning Problem

The data model of the path planning problem can be represented by a 5-tuple:

$$(\boldsymbol{\chi}, \boldsymbol{\chi}_{\text{obs}}, x_{\text{sou}}, x_{\text{tar}}, r), \tag{1}$$

where

1) $\boldsymbol{\chi} \subset \mathbb{R}^n$ is the *state space*, usually $n \in \{2, 3\}$;
2) $\boldsymbol{\chi}_{\text{obs}} \subset \boldsymbol{\chi}$ is the *obstacle state space*;
3) $x_{\text{sou}} \in \boldsymbol{\chi} \setminus \boldsymbol{\chi}_{\text{obs}}$ is the *source*;
4) $x_{\text{tar}} \in \boldsymbol{\chi} \setminus \boldsymbol{\chi}_{\text{obs}}$ is the *target*;
5) r is the radius of the *target.*

Moreover, $\boldsymbol{\chi}_{\text{fre}} = \boldsymbol{\chi} \setminus \boldsymbol{\chi}_{\text{obs}}$ is the *obstacle-free state space.* Let $\boldsymbol{\chi}_{\text{tar}} = \{x \in \boldsymbol{\chi}_{\text{fre}} | \|x - x_{\text{tar}}\| < r\}$ be the *target* space. The path planning problem is to find an *obstacle-free feasible continuous path* $\boldsymbol{\sigma} \subset \boldsymbol{\chi}_{\text{fre}}$ from x_{sou} to any $x \in \boldsymbol{\chi}_{\text{tar}}$. That is, we only need to reach the neighborhood of x_{tar}. The path is defined as a sequence

$$\boldsymbol{\sigma} = \sigma_0 \sigma_1 \ldots \sigma_m, \tag{2}$$

where $\sigma_0 = x_{\text{sou}}$, $\sigma_m \in \boldsymbol{\chi}_{\text{tar}}$, $\forall \lambda \in [0, 1], \lambda \sigma_i + (1 - \lambda)\sigma_{i-1} \in \boldsymbol{\chi}_{\text{fre}}(i = 1, 2, \ldots, m)$, and the direct connection between σ_i and σ_{i-1} is obstacle-free. In other words, the path is essentially a polyline in the map.

The cost of the path is defined as the length of the polyline, i.e.,

$$c(\boldsymbol{\sigma}) = \sum_{i=1}^{m} \|\sigma_i - \sigma_{i-1}\|, \tag{3}$$

where $\|\cdot\|$ is the l_2 norm of a vector.

The path planning problem is to find an optimal path

$$\boldsymbol{\sigma}^* = \arg\min_{\boldsymbol{\sigma}} c(\boldsymbol{\sigma}). \tag{4}$$

3.2 GAN

GAN consists of two components: a generator and a discriminator. The generator generates an image $G_\theta(z)$ using noise $\mathbf{z}$ from the noise space $\mathbf{z} \in \mathbb{R}^n$. The discriminator generates an image $G_\theta(z)$ using data $\mathbf{x}$ from data space $\mathbf{x} \in \mathbf{X}$.

The generator and discriminator can be thought of as playing a game: The generator is trained to generate samples that are as realistic as possible, while the discriminator is trained to correctly classify samples as real or fake. The generator and discriminator networks are trained alternately, with the generator network trying to fool the discriminator, and the discriminator network trying to classify the samples correctly.

The objective function of Gan can be expressed as:

$$\min_\theta \max_\phi (\mathbb{E}_{\mathbf{x}\sim\mathbf{p}_{\text{data}}}[\log D_\phi(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D_\phi(G_\theta(\mathbf{z})))]) \tag{5}$$

The loss function of Gan can be expressed as:

3.3 RRT*

Among the sampling-based algorithms, RRT* is one of the representative algorithms that are widely used. RRT* explores the *state space* by expanding space-filling trees

$$T = (\mathbf{N}, r, p) \tag{6}$$

where

1) $\mathbf{N}$ is the set of all nodes in the tree;
2) $r \in \mathbf{N}$ is the root node;
3) $p : \mathbf{N} \to \mathbf{N} \cup \{\phi\}$ is the parent mapping satisfying
   a) $p(r) = \phi$;
   b) $\forall n \in \mathbf{N}, \exists!\ i \geq 0, \text{s.t. } p^{(i)}(n) = r$;
   c) ϕ represents the empty node.

4 The proposed algorithm

In this section, we first describe the implementation of CGRRT. Then, we prove the probabilistic completeness of CGRRT.

4.1 network structure

4.1.1 GAN

Generative Adversarial Networks (GANs) are a class of deep learning models that have received a lot of attention in recent years due to their ability to generate high-quality synthetic images and other media.

GAN consists of two neural networks: a generator network and a discriminator network. The generator network is trained to synthesize new data samples similar

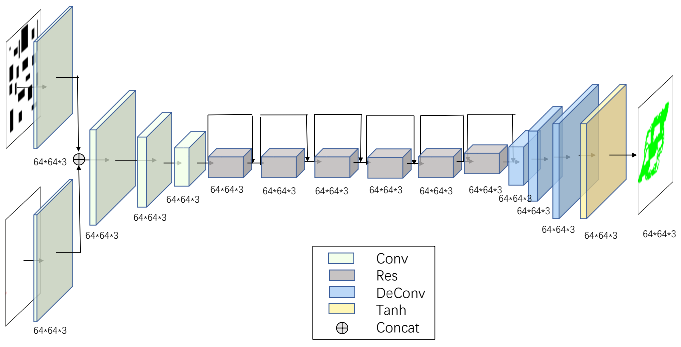


Fig. 1 generator

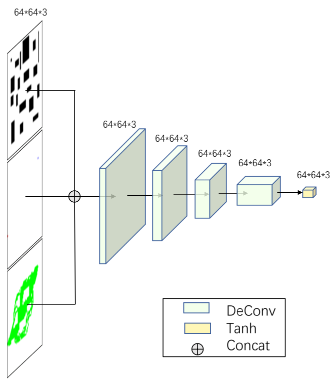


Fig. 2 discriminator

to a given training dataset, while the discriminator network is trained to distinguish real samples from fake samples. The training process of GANs involves an adversarial game between the generator and discriminator networks. Specifically, the generator network is trained to generate samples that are as realistic as possible, while the discriminator network is trained to correctly classify samples as real or fake.

The generator and discriminator networks are trained alternately, with the generator network trying to fool the discriminator, and the discriminator network trying to classify the samples correctly. A key challenge in training GANs is balancing the performance of the generator and discriminator networks.

If the generator network is too weak, it will not be able to synthesize real samples, and if the discriminator network is too weak, it will not be able to effectively distinguish real samples from fake samples. To solve this problem, GANs use specific objective and loss functions to guide the training process. The objective function of GAN is usually defined as a min-max game, where the generator network tries to minimize the loss function, while the discriminator network tries to maximize the loss function. The loss function is usually defined as binary cross-entropy loss, which measures the difference between the predicted label (true or fake) and the ground truth label.

4.2 GAN-RRT*

In this study, we feed the map as an RGB image into a trained CycleGan to predict regions with a high probability of success. The purpose is to mitigate the poor quality and inefficiency of the RRT* algorithm when it first finds a path. To achieve this, we use a GAN to learn areas of paths successfully planned by the RRT* algorithm in the past to predict promising areas in the new map.

After obtaining the predicted promising region, we use the RRT* algorithm for path planning in this region. The RRT* algorithm is a path planning algorithm based on random trees, which can quickly find a reasonable path in a complex environment. In this study, we use the RRT* algorithm to quickly find plausible paths within predicted promising regions. Finally, we take the obtained path as the output of the proposed algorithm.

We hope that the method of this research can improve the efficiency and quality of robot path planning in complex environments.

Algorithm 1 Enhanced RRT* with Clustering and Presearching (ERCP)

```
Input: $\boldsymbol{\chi}, \boldsymbol{\chi}_{\text{obs}}, x_{\text{sou}}, x_{\text{tar}}, \boldsymbol{\chi}_{\text{opt}}, r$
Output: $T$
1: $\mathcal{M} \leftarrow Generator(\boldsymbol{\chi}, \boldsymbol{\chi}_{\text{obs}}, x_{\text{sou}})$;
2: $T = (\mathbf{N}, r, p) \leftarrow (\emptyset, x_{\text{sou}}, \emptyset)$;
3: $\boldsymbol{\chi}_{\text{tar}} \leftarrow \{x \in \boldsymbol{\chi}_{\text{fre}} | \|x - x_{\text{tar}}\| < r\}$;
4: for $i \in \{1, 2, \ldots, k\}$ do
5:     $x_{\text{ran}} \leftarrow SampleFrom(\boldsymbol{\chi}_{\text{opt}})$;
6:     $x_{\text{nst}} \leftarrow Nearest(x_{\text{ran}})$;
7:     $x_{\text{new}} \leftarrow Steer(x_{\text{nst}}, x_{\text{ran}})$;
8:     if $obstacleFree(x_{\text{nst}}, x_{\text{new}})$ then
9:         $\mathbf{N} \leftarrow \mathbf{N} \cup x_{\text{new}}$;
10:        $p(x_{\text{new}}) = x_{\text{nst}}$;
11:        $X_{\text{nea}} \leftarrow NearestNeighborSearch(T, r)$;
12:        $T \leftarrow Rewiring(X_{\text{nea}}, x_{\text{new}})$;
13:        if $x_{\text{sou}} \in \boldsymbol{\chi}_{\text{tar}}$ then
14:            Return $T$;
15: Return $failure$;
```

5 Experiments

In terms of dataset, we use 20 initial maps and generate 200 different maps through data augmentation. Then for each map, randomly select 100 pairs of start and end points, use RRT to find feasible paths among them, 50 successful paths form the promising areas of the map, and finally generate 20,000 sets of data.

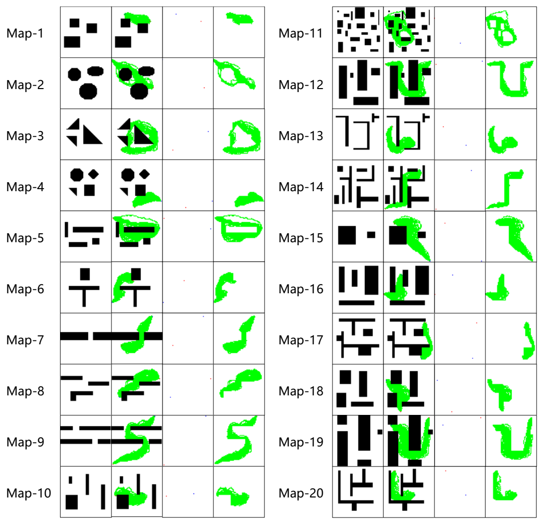


Fig. 3 A display of the original map. On the left is the map name. The first column is the original map. The second column is the data including the starting point and ending point, and the promising area. The third column is the individual start and end points. The fourth column is for individual promising regions. The black parts are obstacles. The blue dot is the starting point and the red dot is the end point. Green areas are promising areas.

In terms of training, we use 16,000 sets of data as training tests, the ratio of training and testing is 4:1, and another 4,000 sets of untrained data are used to evaluate the generalization ability of the model.

In terms of evaluating model performance, we compare our model with the three most popular algorithms (RRT*, IRRT*, SaGan). Among them, IRRT* realizes the heuristic search of ellipse in the path planning process, which speeds up its convergence to the optimal solution. SaGan learns the features of maps and promising regions through residuals and attention, thus giving promising regions between different starting and ending points.

We use windows10 as the training and testing platform of the model, the CPU is AMD Ryzen-7 4800H, the graphics card is RTX 2060 6g, and the memory is 16g. In terms of software, the model implementation language is python3.8, and the deep learning framework is pytorch 1.12.1.

5.1 Dataset

We generated 20 maps of varying complexity, as shown in the figure. In order to enhance the generalization ability of the model, each map translates and rotates the obstacles to generate 9 extended maps. The final map dataset is 200 maps.

By randomly selecting the start and end points in the map, use RRT to perform 50 random iterations in this map, and collect paths that successfully connect the start and end points to form promising regions.

As shown, it shows a set of data for each map. On the far left of the map is the name of the map. The first column is a map with only obstacles, the black parts are obstacles. The second column contains obstacles, start and end points, and promising areas. The blue dot is the start point and the red dot is the end point. Green areas are promising areas. The third column of pictures is the start and end points extracted separately from the picture on the left. The pictures in the fourth column are promising regions.

The purpose of extracting the starting point, ending point and promising area separately is to facilitate data processing.

5.2 Training Details

In all 20,000 datasets, the number of training and testing datasets is 16,000, and the ratio of training and testing sets is 4:1. The number of training rounds is 10 rounds, the learning rate of the generator is 0.0001, the learning rate of the discriminator is 0.00005, and the batch size is 16.

5.3 Evaluate model performance

The quality of the generated promising regions is the key to evaluate the performance of our model. We will evaluate the model in this paper from two aspects. One is to judge the quality of the generated promising regions through connectivity, that is, whether the promising regions can continuously connect the starting point and the ending point. On the other hand, it is the generalization ability of the model. We will use the four untrained base maps, input them into the model, and judge the generalization ability through connectivity.

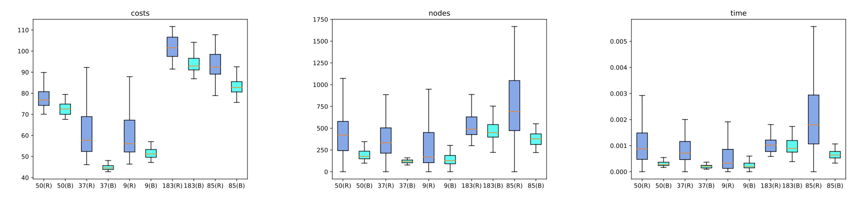
After testing, in the trained map, using the promising regions obtained from the test set, the overall connectivity is 95.08%.

5.4 Initial path quality

As a path planning problem, we evaluate the performance of the algorithm from three main indicators and two aspects. The main indicators include the path cost, the number of sampling nodes, and the time consumed. Two aspects include the main indicators of the initial path and the optimal path. We will conduct multiple experiments on different maps to demonstrate the superiority of our algorithm.

Figure 4 shows the quality of the initial path obtained on different maps, the number of samples and the time consumed.

Figure 4(a) shows the quality of the initial path, the horizontal axis represents different maps, and the vertical axis represents the quality of the path. The boxplots in light blue represent RRT* and bright green represent our algorithm. It can be seen from the figure that the path quality of B-RRT* is overall better than that of RRT*.



(a) costs (b) nodes taken. (c) time.

Fig. 4 Illustration of initial path quality.

5.5 Optimal Path Quality

5.6 Discussion

6 Conclusion and further work

Acknowledgements

Data availability statement

Data will be available upon reasonable request.

References

1. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik **1**(1) (1959) 269–271
2. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics **4**(2) (1968) 100–107
3. LaValle, S.M., Kuffner, J.J.: Rapidly-exploring random trees: Progress and prospects. International Journal of Robotics Research **20**(5) (2001) 378–400
4. Karaman, S., Frazzoli, E.: Sampling-based Algorithms for Optimal Motion Planning. The International Journal of Robotics Research **30**(7) (2011) 846–894
5. Kuffner, J.J., LaValle, S.M.: Randomized kinodynamic planning. In: IEEE International Conference on Robotics and Automation, IEEE (2000) 1277–1283