IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Review on Collision Avoidance and Trajectory Planning In Autonomous Systems

Ashwin R
Department of Robotics And Automation
B.E Robotics And Automation
PSG College of Technology
Coimbatore

Balaji J
Department of Robotics And Automation
B.E Robotics And Automation
PSG College of Technology
Coimbatore

Dr.Anbarasi MP
Department of Robotics And Automation
B.E Robotics And Automation
PSG College of Technology
Coimbatore

***Abstract*— Tone- driving buses, UAVs, and indeed mobile robots have been the focus of a lot of exploration in recent times. To make sure these systems are safe and effective, it's important to know how to design and use collision avoidance and line planning algorithms. Then is a quick look at some of the crucial generalities and principles that go into these systems. The thing of line planning is to produce a path for independent systems to follow while trying to reach different pretensions, like reducing trip time, saving energy, or keeping them down from obstacles. It's important to understand the system's dynamic dynamics, limits, and terrain, like static obstacles and moving objects, to make sure the system can navigate safely. The collision avoidance algorithms use detector data, like Lidar or Radar, to identify and track objects in the terrain, and also make opinions in real- time to acclimate the line of the system to avoid collisions.**

Keywords—collision avoidance, planning algorithms, path planning

I. INTRODUCTION

In the realm of autonomous systems, be it tone- driving buses , unmanned upstanding vehicles(UAVs), or mobile robots, the capability to navigate safely and efficiently through dynamic surroundings is of consummate significance. This necessitates the operation of sophisticated algorithms for line planning, collision avoidance, and path planning. These algorithms are abecedarian in orchestrating the movement of independent systems, icing they can reach their objects while avoiding obstacles and hazards.

Line planning is a central conception in this sphere. It involves the creation of a doable path for an independent system to cut , taking into account a multitude of variables and objects. These objects may encompass minimizing trip time, conserving energy, or maintaining a safe distance from obstacles. A critical aspect of line planning is the understanding of the system's dynamics, limitations, and its commerce with the terrain, which includes factors like the presence of static obstacles and the dynamics of moving objects.

Completing line planning, collision avoidance algorithms are necessary in precluding potentially dangerous hassles with objects or obstacles in the terrain. These algorithms calculate on detector data, similar as Lidar, Radar, or cameras, to descry and track these objects. Once linked, collision avoidance algorithms make real- time opinions to acclimate the system's line, effectively steering it down from implicit collisions.

Path planning, nearly affiliated to line planning, involves the selection of a sequence of waypoints or locales that the independent system should cut to reach its ultimate destination. Path planning considers the system's starting point, thing position, and the configuration of the terrain to determine the most effective and handicap-free route.[1]

II. VITAL REQUIREMENT

A. *Autonomous Systems*

The need for autonomous systems is driven by a variety of factors that gauge safety, effectiveness, and productivity. Autonomous systems have the eventuality to significantly enhance safety by reducing accidents caused by mortal error, particularly in the case of independent vehicles, where distracted or disabled driving is a common concern. also, these systems can boost effectiveness by automating tasks in manufacturing, reducing labor costs, and adding productivity. In diligence facing labor dearths or demanding work conditions, similar as husbandry and construction, independent systems offer a result by performing repetitious or physically demanding tasks. also, independent systems contribute to cost reduction, as they can work continuously without fatigue. They also offer environmental benefits by optimizing energy consumption and reducing emigrations, particularly in transportation where they can ameliorate energy effectiveness and business operation. As technology advances, independent systems find operations in space and deep- ocean disquisition, healthcare, disaster response, and multitudinous other disciplines, effectively addressing a wide range of societal and artificial requirements.

B. *Collision Avoidance and Trajectory Planning*

Collision avoidance and line planning are vital for independent systems, icing safety, effectiveness, and rigidity. They continuously assess the terrain, minimizing accident pitfalls in operations like tone- driving buses and optimizing paths for effective navigation, reducing trip time and enhancing energy effectiveness. These algorithms acclimatize to dynamic surroundings, avoiding obstacles, whether stationary or moving. In resource- constrained scripts, they optimize resource operation. Precision and pungency are pivotal for operations like surgical robots and artificial robotization, reducing mortal intervention. As independent systems gain, collision avoidance and line planning come decreasingly critical, enabling safe and effective technology integration.[2]

III. TYPES OF PATH PLANNING

There are many methods for the AUV path planning. According to the basic principles of the algorithm, the path planning algorithms are divided into four categories, including artificial potential field methods, geometric model search methods, random sampling methods, and intelligent bionic methods.[3]

A. Artifical Potential Field Methods

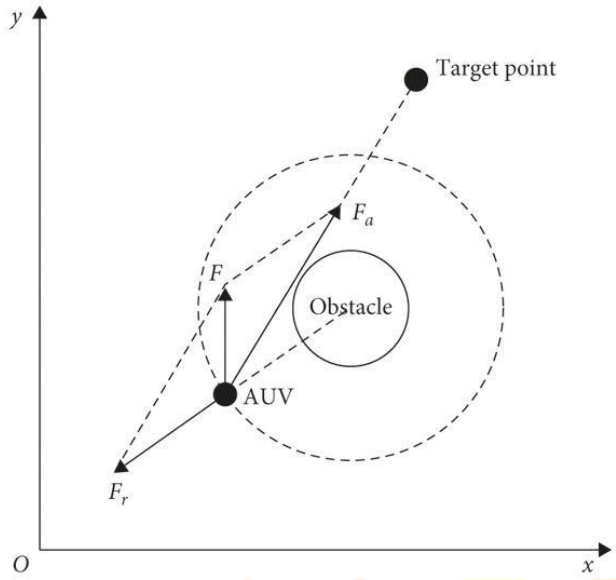


Fig.1. Diagram For Artificial Potential Field Method

The Artificial Implicit Field system is a extensively used approach in robotics and independent systems for path planning and handicap avoidance. It simulates seductive and repulsive forces, akin to the way objects interact in the physical world, to guide independent systems through complex surroundings, as shown in Fig.1 seductive forces pull the system toward its thing, while repulsive forces push it down from obstacles, determining its line grounded on the net force. This system is valued for its simplicity and computational effectiveness, enabling real- time rigidity to dynamic surroundings. still, it does have limitations, similar as the eventuality to get stuck in original minima or induce oscillations in narrow passages. To address these issues, more advanced styles like the Vector Field Histogram and the Elastic Band system have been developed, erecting upon the implicit field conception. The Artificial Implicit Field system finds operations in robotics, drone path planning, and indeed videotape game character movement simulations, particularly in scripts taking real- time navigation.[4]

B. Geometric Model Search Methods

Geometric models and search styles are pivotal for assaying 3D data, enabling effective disquisition and understanding of complex surroundings. Geometric models, like point shadows and morass, give structured representations of shapes and spatial connections. Meanwhile, search styles are essential algorithms for data reclamation and analysis within these models. They play crucial places in computer plates, robotics, and medical imaging. Whether relating collisions in virtual surroundings, enabling path planning for independent systems, or abetting medical judgments through anomaly discovery, these tools bridge the gap between the physical and digital 3D world, easing the creation of virtual surroundings, structural design, and problem- working across different disciplines. The flow diagram of the geometric model search method is as shown in fig.2 [5]

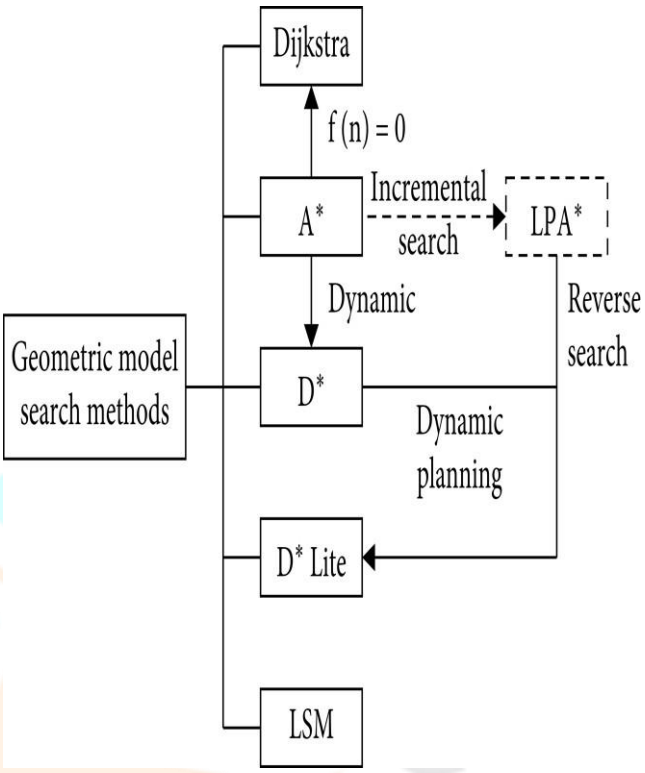


Fig.2. Process of Geometric Model Search Methods

C. Random Sampling Methods

Random slice styles, similar as the fleetly exploring arbitrary tree(RRT) system and the probabilistic roadmap(PRM) system, perform well in the AUV path planning. This type of system uses arbitrary slice to make probabilistic discovery of the AUV aquatic working terrain to search the path, but this probability or randomness brings some challenges to the AUV path planning. Flow chart is as shown in Fig.3

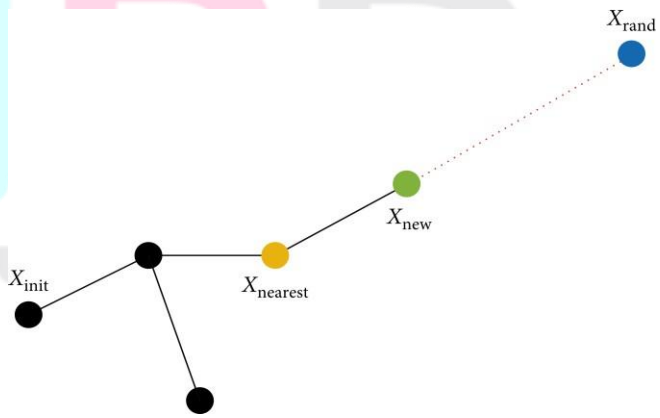


Fig.3. Flow Chart of Random Sampling Method

D. *Intelligent Bionic Methods*

Intelligent bionic systems combine biology and technology, aiming to replicate or enhance natural capabilities. They draw alleviation from the natural world, like mimicking beast movements or replicating brain functions. These systems find operations in healthcare, furnishing advanced prosthetics, neural interfaces, and medical bias.

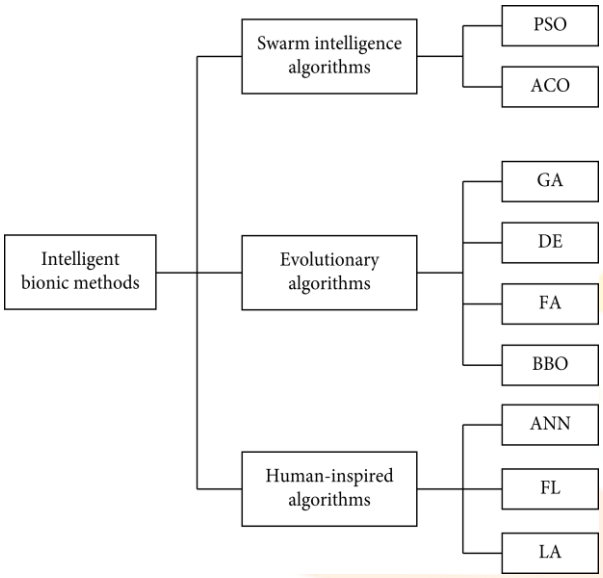


Fig.4. Flow chart of Intelligent Bionic Method

Bionics also prop environmental monitoring and conservation. In robotics, biomimetic designs ameliorate hunt and deliverance operations and robotization. still, they pose challenges, similar as ethical enterprises about mortal-technology integration and implicit impacts on employment. In summary, intelligent bionic systems offer instigative openings in colorful disciplines, reshaping the future of mortal- machine integration.Fig.4 [6]

IV. COLLISION AVOIDANCE

Collision avoidance is a vital conception in independent systems, robotics, and transportation, aiming to help physical collisions and enhance safety. It relies on real- time monitoring, detector data, and nippy decision- making to avoid collisions. operations gauge tone- driving vehicles, aeronautics, maritime transport, and worker safety in diligence. Yet, challenges live, similar as the need for real-time data and robust decision- timber. Technological advancements, including AI and V2V communication, have bettered these systems. Collision avoidance not only ensures safety but also boosts functional effectiveness, optimizing routes and minimizing accidents. In an connected world, this technology continues to play a vital part in securing lives and means.

A. *Mathematical Principle*

The number of robot joints is n, which can be expressed as (J_1,J_2,J_3……J_n). The configuration space of the robot is $Q \subset R^n$. The subset of Q composed of collision-free configurations is collision-free configuration space Q_{col_free}.

The path points number on the trajectory is m, so the robot trajectory can be expressed as $\xi \in \mathbb{R}^{mn}$:

$$\xi = (\underbrace{q_1^0, q_1^1, \ldots, q_1^m}_{J_1}, \underbrace{q_2^0, q_2^1, \ldots, q_2^m}_{J_2}, \ldots, \underbrace{q_n^0, q_n^1, \ldots, q_n^m}_{J_n}) \qquad (1)$$

Because the method treats collision avoidance and task constraints as hard constraints rather than optimization terms, the cost function only contains the smooth term $f_{smooth}(\xi)$ to measure dynamical quantities across the trajectory. $f_{smooth}(\xi)$ can be precisely computed by a sum of squared derivatives.

For each joint's trajectory ξ_i, $i \in \{1, 2, \ldots, n\}$, a finite differential matrix $\mathbf{K}$ can be constructed as:

$$\mathbf{K} = \begin{bmatrix} 1 & 0 & 0 & & 0 & 0 & 0 \\ -2 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & -2 & 1 & & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & 0 & & 1 & -2 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 & -2 \\ 0 & 0 & 0 & & 0 & 0 & 1 \end{bmatrix} \otimes I_{m\times m} \qquad (2)$$

Which will make :

$$\ddot{\xi}_i = \mathbf{K}\xi_i \qquad (3)$$

$$\ddot{\xi}_i^{\mathrm{T}}\ddot{\xi}_i = \xi_i^{\mathrm{T}}(\mathbf{K}^{\mathrm{T}}\mathbf{K})\xi_i = \xi_i^{\mathrm{T}}\mathbf{R}\xi_i \qquad (4)$$

where $\mathbf{R} = \mathbf{K}^{\mathrm{T}}\mathbf{K}$, and $\xi^{\mathrm{T}}_i\,\mathbf{R}\xi_i$ represents the sum of squared accelerations along the ith joint's trajectory.

In order to calculate the weighted sum of the smooth cost of all joints' trajectories, we construct a positive Hessian matrix $\mathbf{H}$:

$$\mathbf{H} = \begin{bmatrix} w_1\mathbf{R} & 0 & \cdots & 0 \\ 0 & w_2\mathbf{R} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_n\mathbf{R} \end{bmatrix} \qquad (5)$$

where w_i is the weight of the joint J_i.

In this way, the optimization object can be formulated as:

$$U(\xi) = \frac{1}{2}\sum_{i=1}^{n} w_i \left\|\xi_i\right\|_{\mathbf{R}}^2 = \frac{1}{2}\xi^{\mathrm{T}}\mathbf{H}\xi \qquad (6)$$

Through a first order Taylor expansion, the optimization object can be expressed as:

$$U(\xi) \approx U(\xi_k) + g_k^{\mathrm{T}}(\xi - \xi_k) \qquad (7)$$

where g_k is the gradient of the optimization object at ξ_k:

$$g_k = \nabla(U(\xi_k)) = \nabla(\tfrac{1}{2}\xi_k{}^{\mathrm{T}}\mathbf{H}\xi_k) = \mathbf{H}\xi_k \qquad (8)$$

Further, the update rule after adding regular term can be expressed as:

$$\xi_{k+1} = \arg\min_{\xi}\left\{U(\xi_k) + g_k^{\mathrm{T}}(\xi - \xi_k) + \tfrac{\lambda}{2}\|\xi - \xi_k\|_M^2\right\} \qquad (9)$$

Let $\Delta\xi = \xi - \xi_k$, then:

$$\Delta\xi = \arg\min_{\Delta\xi}\left\{U(\xi_k) + g_k^{\mathrm{T}}\Delta\xi + \tfrac{\lambda}{2}\|\Delta\xi\|_{\mathbf{M}}^2\right\} \qquad (10)$$

where the notation $\|\Delta\xi\|2M$ denotes the norm of displacement between the current trajectory and the updated trajectory with respect to the Riemannian metric M. λ is a normalizing factor to balance the trajectory smoothness and the updating of step length.

Therefore, the trajectory updating process without constraints can be expressed as:

$$xi_{k+1} = \xi_k + \alpha_k \Delta\xi_k \qquad (11)$$

where α_k is the update rate of the *k*th iteration.

So far, we have deduced the optimization process without considering collision avoidance and task constraints.[7]

B. Collision Bactrack

The initial trajectory ξ_0 which needs to be post process is totally collision-free, and most of SBMPs can guarantee this:

$$\mathbf{q} \in Q_{col\_free}, \quad \forall \mathbf{q} \in \xi_0 \qquad (12)$$

But the changes to the trajectory caused by the optimization process may lead to violation of the collision-free state. So we propose the collision backtrack mechanism to convert the collision avoidance condition into linear constraints of quadratic programming.

Assume that after *i* iteration, the trajectory is still collision-free. In *i* + 1 iteration, a collision occurs at τ[τ ∈ (0, 1)] moment, and the collision points on the two rigid bodies are $\mathbf{P}_1$ and $\mathbf{P}_2$, respectively. We backtrack to the previous collision-free trajectory ξ_i at τ moment, and the two states are shown in Fig.5.

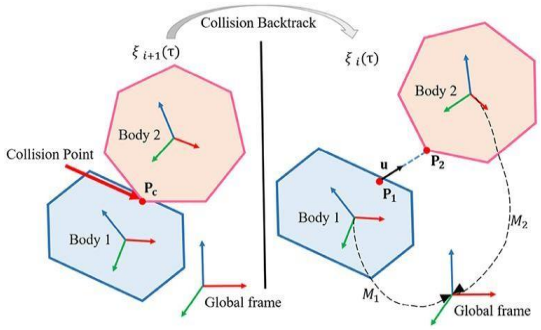


Fig.5.Collision Backtrack

The unit vector between two points P1, P2 is defined as u:

$$\mathbf{u} = \frac{M_2^1(\xi_i(\tau))\mathbf{P}_2 - \mathbf{P}_1}{\|M_2^1(\xi_i(\tau))\mathbf{P}_2 - \mathbf{P}_1\|} \qquad (13)$$

where $M^1_2(q) = M_1(q)^{-1}M_2(q)$, is the transformation matrix between the two rigid bodies when the robot configuration is *q*, which can be solved by positive kinematic.

In order to ensure there is no more collision at this position in future optimization, we should forbid $\mathbf{P}_1$ and $\mathbf{P}_2$ to move toward each other. Mathematically, the projection of the relative position change vector $\Delta\mathbf{x}$ of the two points on $\mathbf{u}$ should be ≥0. As shown in fig 6, we discuss the two situations according to the location of the collision points.

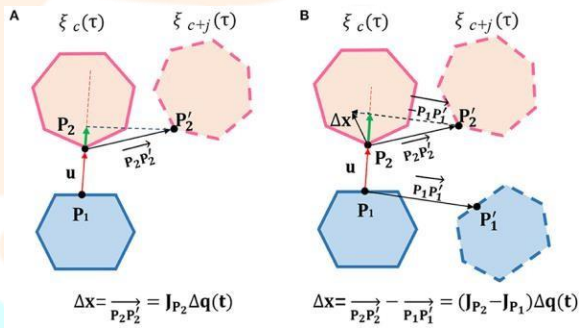


Fig.6.Calculation of Collison Backtrack

If one of the collision rigid bodies is on the robot, the other is on a static obstacle:

$$\mathbf{u}^{\mathrm{T}}\mathbf{J}_{\mathbf{P}_2}\Delta\mathbf{q}_\tau = \mathbf{u}^{\mathrm{T}}\mathbf{J}_{\mathbf{P}_2}\mathbf{X}_\tau(\xi_{c+j} - \xi_c) \geq 0 \qquad (14)$$

where $\mathbf{J}_{P2}$ is a 3 × *n* Jacobian matrix, and $\Delta\mathbf{q}_\tau$ is the joints increment of the robot between $\xi_c(\tau)$ and $\xi_{c+j}(\tau)$.

To extract the robot configuration at τ moment, a *n* × *mn* sparse matrix $\mathbf{X}_\tau$ is constructed. Since the path is a combination of piecewise linear sub-paths, we can find the two path points $\mathbf{q}^k$ and $\mathbf{q}^{k+1}$ adjacent to τ in the time dimension. There exists a β ∈ [0, 1] that enables ξ(τ) to be written in linear combination of two path points:

$$\xi(\tau) = \mathbf{q}^k + \beta(\mathbf{q}^{k+1} - \mathbf{q}^k) = (1-\beta)\mathbf{q}^k + \beta\mathbf{q}^{k+1}, \beta \in [0,1] \quad (15)$$

$$\mathbf{B} = \left[\underbrace{0, \cdots, 0, 1-\beta}_{k}, \beta, 0, \cdots, 0\right]_{1\times m} \quad (16)$$

For every joint of the robot:

$$\mathbf{X}_\tau = \begin{bmatrix} \mathbf{B} & 0 & \cdots & 0 \\ 0 & \mathbf{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{B} \end{bmatrix}_{n\times mn} \quad (17)$$

Similarly, if the two collision rigid bodies are both on robot, the linear constraint can be expressed as:

$$\mathbf{u}^{\mathrm{T}}(\mathbf{J}_{\mathbf{P}_2} - \mathbf{J}_{\mathbf{P}_1})\Delta\mathbf{q}_\tau = \mathbf{u}^{\mathrm{T}}(\mathbf{J}_{\mathbf{P}_2} - \mathbf{J}_{\mathbf{P}_1})\mathbf{X}_\tau(\xi_{c+j} - \xi_c) \geq 0 \quad (18)$$

Extract the known term of the Equation (14) and (18):

$$\Phi = \begin{cases} \mathbf{u}^{\mathrm{T}}(\mathbf{J}_{\mathbf{P}_2} - \mathbf{J}_{\mathbf{P}_1})\mathbf{X}_\tau, & \mathbf{P}_1, \mathbf{P}_2 \text{ both on robot} \\ \mathbf{u}^{\mathrm{T}}\mathbf{J}_{\mathbf{P}_2}\mathbf{X}_\tau, & \mathbf{P}_2 \text{ on robot} \end{cases} \quad (19)$$

Then the linear constraints for quadratic programming can be expressed as:

$$\mathbf{C} = \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_k \end{bmatrix} \Rightarrow \mathbf{C}\Delta\xi \geq \mathbf{0} \quad (20)$$

In addition to collision avoidance constraints, the start and ending points of the trajectory should be fixed. We use matrix **D** to extract the start and ending points of the strajectory:

$$\mathbf{D} = [1.0,\ 0,\ 0,\ \cdots,\ 0,\ 0,\ 1.0]_{1\times m} \quad (21)$$

$$\mathbf{S} = \begin{bmatrix} \mathbf{D} & 0 & \cdots & 0 \\ 0 & \mathbf{D} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{D} \end{bmatrix}_{n\times mn} \quad (22)$$

Then the constraint condition can be expressed as:

$$\mathbf{S}\Delta\xi = \mathbf{0} \quad (23)$$

Ignoring the constant terms of Equation (10), and combining the constraints of Equations (20, 23), the linear constrained Quadratic programming (QP) problem for trajectory optimization can be formulated as following:[7]

$$\min_{\Delta\xi} \quad \frac{\lambda}{2}\Delta\xi^{\mathrm{T}}\mathbf{M}\Delta\xi + g^{\mathrm{T}}\Delta\xi \qquad \mathbf{C}\Delta\xi \geq \mathbf{0} \qquad \mathbf{S}\Delta\xi = \mathbf{0}. \quad (24)$$

C. Algorithm

i. LCQP

In this section the process of trajectory optimization by linear constrained quadratic programming (LCQP) will be explained in the form of codes.

As shown in fig 7, the input of the algorithm is a collision-free trajectory generated by SBMPs (sampling-based motion planning). The output is the optimized trajectory by LCQP.

Input: Initial collision-free trajectory needs to be optimized ξ_0
Output: Optimized collision-free trajectory ξ_0

```
1  ξ0 ← randomShortcut(ξ0);
2  initialConstraints(S);
3  while True do
4      Δξ ← QPIterate(ξ0, C, S);
5      if (‖Δξ‖ < 10^-3 and !collosion) then
6          return ξ0;
7      end
8      ξ1 ← ξ0 + α Δξ;
9      jointProjection(ξ1);
10     manifoldProjection(ξ1, B_t);
11     if (PathValidate(ξ1)) then
12         ξ0 ← ξ1;
13         collision ← False;
14     end
15     else
16         collision ← True;
17         Φ ← computeCollisionConstraint(ξ0, ξ1);
18         C ← addToLinearConstraints(Φ);
19     end
20 end
```

Fig.7.Algorithm for LCQP

Lines 3 to 20 are the process of trajectory optimization. Based on the current linear constraints **C** and **S**, the incremental of the trajectory $\Delta\xi$ can be calculated by the LCQP which is formulated as Equation (24). If the incremental of trajectory is small enough and the previous trajectory ξ_0 is totally collision-free, ξ_0 will be returned as the optimization result. Otherwise, the candidate trajectory ξ_1 will be updated according to Equation (11).[7]

ii. RRT

The Rapidly- Exploring Random Tree(RRT) algorithm is a critical stir planning algorithm in robotics and independent systems. Its primary thing is to find a doable

path from a starting point to a thing within a complex and dynamic terrain while avoiding obstacles. RRT operates by iteratively growing a tree structure, with each knot representing a system state in the configuration space.

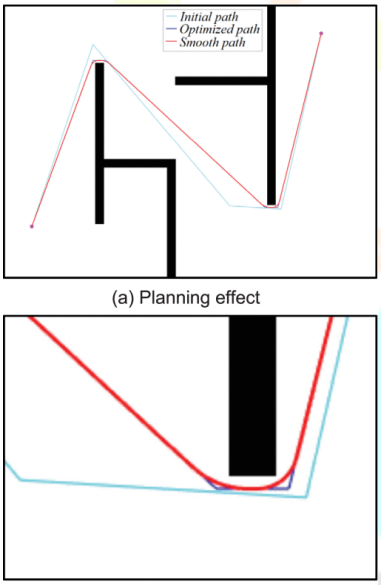
In each replication, a arbitrary knot is tried and connected to the being tree, with a bias towards the thing direction. The algorithm ensures handicap avoidance by checking for collisions before adding a new knot to the tree. RRT's operation spans colorful fields, including robotics, drone path planning, and videotape game design, due to its rigidity to dynamic and high- dimensional spaces.

Extensions like RRT * and RRT- Connect enhance its capabilities. still, it may not always guarantee an optimal path, and it can be computationally demanding in high-dimensional spaces. In summary, RRT is a protean tool for stir planning, balancing disquisition and confluence in complex surroundings.

V. EXPERIMENT

2D Experiment

As shown in Fig 9, we use a different step length λ for the RRT algorithm, and use the proposed method to optimize the trajectory produced by RRT. The size of the 2D maps are 1 × 1 rad^2 which means the joint limit is set as 0 rad to 1 rad. Velocity and acceleration limits used to calculate the execution time are set to 1.2 rad/s and 1.5π rad/s^2, respectively.



(a) Planning effect

(b) Path details

Fig.9.2D Experiment

VI. CONCLUSION

In conclusion, recent times have seen a lot of interest around the use of autonomous vehicles like self-driving buses, UAVs, and mobile robots. What's at the heart of ensuring their success is understanding how to make them safe and effective, which brings us to the world of collision avoidance and line planning algorithms.

Line planning, in simple terms, is all about finding the best path for these machines to follow while they're trying to meet different goals. It could be about getting to a destination faster, saving energy, or avoiding obstacles on their way. To achieve this, we must take into account the system's behaviour in different situations, its limits, and the lay of the land, which includes both stationary obstacles and things on the move.

Now, let's talk about collision avoidance. Think of it as the real-time reflexes of these systems. They use gadgets like Lidar and Radar to keep an eye on the surroundings, just like we use our senses. If they spot something in their way, they make quick decisions to steer clear and avoid accidents. It's almost like the "auto-pilot" of our vehicles but way more sophisticated.

So, these systems are like intelligent drivers on the road, and they rely on these algorithms to keep them and everyone else safe. In a world where autonomous vehicles are becoming more common, understanding and perfecting these technologies is essential to a smooth and secure ride into the future. “Collision avoidance and trajectory planning in autonomous systems” emphasizes the importance of reliable navigation. These systems are essential in diverse fields such as tone-driving vehicles, drones and artificial robots where the importance of collision avoidance and trajectory planning is paramount for their safety and trustworthiness. This paper examines the intricacies of these algorithms, highlighting their rigidity in dynamic and complex environments. It emphasizes the need for accurate and flexible navigation in face of unanticipated challenges. Advances in these technologies demonstrate our ability to leverage artificial intelligence (AI) and machine literacy, transforming transportation, trip and relations with our environment. However, the journey toward fully autonomous systems continues, requiring continued research and collaboration.

REFERENCES

[1] Gerdts, M., Henrion, R., Hömberg, D. and Landry, C., 2011. Path planning and collision avoidance for robots.

[2] Almasri, M.M., Alajlan, A.M. and Elleithy, K.M., 2016. Trajectory planning and collision avoidance algorithm for mobile robotics system. *IEEE Sensors journal*, *16*(12), pp.5021-5028.

[3] Guo, Y., Liu, H., Fan, X. and Lyu, W., 2021. Research progress of path planning methods for autonomous underwater vehicle. *Mathematical Problems in Engineering*, *2021*, pp.1-25.

[4] Fan, X., Guo, Y., Liu, H., Wei, B. and Lyu, W., 2020. Improved artificial potential field method applied for AUV path planning. *Mathematical Problems in Engineering*, *2020*, pp.1-21.

[5] Zhang, B., Liu, W., Mao, Z., Liu, J. and Shen, L., 2014. Cooperative and geometric learning algorithm (CGLA) for path planning of UAVs with limited information. *Automatica*, *50*(3), pp.809-820.

[6] Guo, Y., Liu, H., Fan, X. and Lyu, W., 2021. Research progress of path planning methods for autonomous underwater vehicle. *Mathematical Problems in Engineering*, *2021*, pp.1-25.

[7] Liu, Y., Zha, F., Li, M., Guo, W., Jia, Y., Wang, P., Zang, Y. and Sun, L., 2021. Creating Better Collision-Free Trajectory for Robot Motion Planning by Linearly Constrained Quadratic Programming. *Frontiers in Neurorobotics*, *15*, p.724116.

[8] Wang, B., Ju, D., Xu, F., Feng, C. and Xun, G., 2022. CAF-RRT*: A 2D Path Planning Algorithm Based on Circular Arc Fillet Method. *IEEE Access*, *10*, pp.127168-127181.