分类号：TP242.6　　学校代码：10363
密　级：公开　　学　　号：2220320125

安徽工程大学
Anhui Polytechnic University

硕士学位论文

题目　基于改进 RRT*-DWA 混合算法的智能车路径规划算法研究

论 文 作 者　于卓越
校 内 导 师　韩超（教授）
校 外 导 师　郝旭耀
专业学位类别　电子信息
研究方向（领域）　人工智能

论文提交日期：2025 年 5 月 16 日

分类号：TP242.6　　单位代码：10363
密级：公开　　学　号：2220320125

题　　目　基于改进 RRT*-DWA 混合算法的智能车路径规划算法研究

英文并列

题　　目　Research on intelligent vehicle path planning algorithm based on improved RRT*-DWA hybrid algorithm

学　生　姓　名　：　于卓越
校　内　导　师　：　韩超（教授）
校　外　导　师　：　郝旭耀
专　业　（　领　域　）：　电子信息
研 究 方 向 （ 领 域 ）：　人工智能

论文答辩日期：2025 年 5 月 12 日

安徽工程大学学位论文原创性声明

本人郑重声明：我恪守学术道德，崇尚严谨学风。所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已明确注明和引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品及成果的内容。论文为本人亲自撰写，我对所写的内容负责，并完全意识到本声明的法律后果由本人承担。

学位论文作者签名： 

日期：2025 年 5 月 16 日

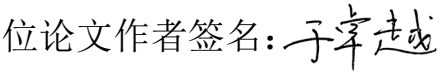
安徽工程大学学位论文版权使用授权书

学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅或借阅。本人授权安徽工程大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保密□，在　　年解密后适用本版权书。

本学位论文属于

不保密 ☑。

学位论文作者签名：  　　指导教师签名： 

日期：　2025 年　5　月　16 日　　日期：2025　年　5　月　16 日

基于改进 RRT*-DWA 混合算法的智能车路径规划算法研究

摘 要

随着科技的发展，智能车作为集智能感知、路径规划、动态决策等多种功能于一体的新一代自主移动技术平台，在社会发展中扮演着越来越重要的角色。路径规划作为智能车的核心技术，已成为该领域的研究重点。然而，随着智能车的应用场景日渐复杂，如何在规划可行路径的同时兼顾实时性与安全性已成为路径规划算法亟待解决的关键问题。本文以快速扩展随机树算法（Rapidly-exploring Random Tree Star algorithm, RRT*）和动态窗口法（Dynamic Window Approaches, DWA）为基准算法，针对复杂环境下的路径规划问题展开研究，主要工作和研究内容如下：

（1）针对全局路径规划算法在复杂环境中收敛速度慢、路径代价高等问题，提出一种基于精确采样的快速扩展随机树算法(Precise Sampling Rapidly-exploring Random Tree Star, PSRRT*)。首先，通过精确采样模型生成目标导向性强、分布合理的采样点，加快算法的收敛速度；其次，引入自适应节点扩展模型，动态调整节点的扩展方向，进一步提高算法的全局收敛能力；最后，结合贪婪算法和 B 样条平滑算法对路径进行优化，有效提高了全局路径的整体平滑性并降低了路径代价。仿真实验表明，PSRRT*算法规划的全局路径代价更低、算法收敛速度更快，能够更好地满足复杂环境下的路径规划需求。

（2）为提升 PSRRT*算法在动态环境中的实时性与鲁棒性，本文提出一种基于热传递的双动态窗口法（Bi-Dynamic Windows Approaches based on Heat Transfer, BDWAHT）。首先，建立双动态窗口模型优化评估信息，以减少计算复杂度，提高算法实时性；其次，设计基于热传递的窗口自适应调节机制，以增强算法对不同环境的适应能力；最后，构建新的轨迹评价函数，以解决 PSRRT*算法与 BDWAHT 算法融合过程中路径的波动问题，从而提高规划路径的局部平滑性。仿真实验验证了 PSRRT*-BDWAHT 融合算法在复杂环境中的性能。

（3）基于 ROS 搭建了智能车的仿真实验平台，对 PSRRT*-BDWAHT 融合算法进行实验测试。在仿真测试环节，利用 Gazebo 软件构建包含静态障碍物和动态障碍物的复杂测试环境，以模拟真实场景中的多样化路况。同时，借助 Rviz 可视化工具对路径规划过程进行动态呈现，以便分析该算法在动态避障与路径平滑性等方面的性能表现。为验证 PSRRT*-BDWAHT 融合算法的实际应用能力，设置了真实环境下的实车实验，并通过与其他算法的对比分析，验证了该算法的有效性及可行性。

关键词：路径规划，RRT*算法，DWA 算法，ROS

Research on intelligent vehicle path planning algorithm based on improved RRT*-DWA hybrid algorithm

ABSTRACT

With the development of science and technology, intelligent vehicles, as a new generation of autonomous mobile technology platform integrating intelligent perception, path planning, dynamic decision-making and other functions, play an increasingly important role in social development. Path planning, as the core technology of intelligent vehicles, has become the focus of research in this field. However, with the increasing complexity of the application scenarios of intelligent vehicles, how to plan feasible paths while considering real-time and safety has become a key problem to be solved by path planning algorithms. In this paper, Rapidly-exploring Random Tree algorithm Star (RRT*) and Dynamic Window Approaches (DWA) are taken as the benchmark algorithms to study the path planning problem in complex environments. The main work and research contents are as follows:

(1) Aiming at the problems of slow convergence speed and high path cost of global path planning algorithms in complex environments, a Precise Sampling Rapidly-exploring Random Tree Star (PSRRT*) is proposed. Firstly, the precise sampling model generates goal-oriented and reasonably distributed sampling points to accelerate the convergence speed of the algorithm; secondly, the adaptive node expansion model is introduced to dynamically adjust the expansion direction of the nodes, which further improves the global convergence ability of the algorithm;

finally, the path optimization is performed by combining the greedy algorithm and the B-spline smoothing algorithm, which effectively improves the overall smoothing of the global path and reduces the path cost. Simulation experiments show that the PSRRT* algorithm plans a global path with lower cost and faster convergence, which can better meet the needs of path planning in complex environments.

(2) In order to enhance the real-time and robustness of PSRRT* algorithm in dynamic environment, this paper proposes a Bi-Dynamic Windows Approaches based on Heat Transfer (BDWAHT). Firstly, the Bi-Dynamic Windows model is established to optimize the evaluation information for reduce the computational complexity and improve the real-time performance of the algorithm; secondly, the adaptive adjustment mechanism of the window based on heat transfer is designed to enhance the adaptive ability of the algorithm for different environments; finally, a new trajectory evaluation function is constructed to solve the fluctuation problem of paths in the process of fusion of the PSRRT* algorithm and the BDWAHT algorithm, so as to increase the planning path's local smoothness. Simulation experiments verify the performance of the PSRRT*-BDWAHT fusion algorithm in complex environments.

(3) A simulation experiment platform for intelligent vehicles was built based on ROS to experimentally test the PSRRT*-BDWAHT fusion algorithm. In the simulation test session, Gazebo is used to construct a complex test environment containing static and dynamic obstacles in order to simulate the diverse road conditions in real scenarios. Meanwhile, the path planning process is dynamically presented with the help of Rviz visualization tool in order to analyze the performance of the algorithm in terms of dynamic obstacle avoidance and path smoothness. In order to verify the practical application capability of the PSRRT*-BDWAHT

fusion algorithm, real vehicle experiments in real environments are set up, and the effectiveness and feasibility of the algorithm are verified by comparing and analyzing with other algorithms.

KEY WORDS: Path Planning, RRT* Algorithm, DWA Algorithm, ROS

Zhuoyue Yu (Electronic Information)
Supervised by Professor Chao Han

目录

第1章 绪 论

1.1 课题研究背景及意义

在现代社会中，汽车已成为当今最普遍的交通工具之一，为人们的日常生活带来了极大的交通便利。然而，随着社会经济水平的快速发展，汽车保有量也在逐年上升，交通环境也变得愈发复杂。密集的车流和复杂的道路大幅度提升了事故的发生率。据中国国家统计局发布的《中国统计年鉴 2024》数据显示，2023年全国共发生交通事故 25 万起，死亡人数高达 6 万余人。仅依靠改善道路基础设施和强化交通法律法规难以从根本上解决交通安全问题。因此，研究学者将自动驾驶技术引入汽车提出智能车的概念，以减少因人为操作不当导致的交通事故。为推动智能车行业的发展，我国印发了《智能汽车创新发展战略》，文中明确指出发展智能汽车对我国具有重要的战略意义。智能车不仅应用于出行交通领域，在如图 1-1 所示的新能源汽车、智能仓储、城市服务、医疗护理、农业生产与矿产开发等多个行业领域，也展现出巨大的产业价值[1-7]。智能车在多个行业领域的广泛应用主要归因于其技术体系的日益成熟以及与各领域应用场景的深度融合[8-10]。

（a） 新能源汽车　（b） 智能仓储　（c） 城市服务

（d） 医疗护理　（e） 农业生产　（f） 矿产开发

图 1-1 智能车应用领域

早在 1956 年至 1972 年，美国科学家查理·罗森就研制出世界上第一台智能车—Shakey。尽管从今天的角度来看，Shakey 的设计结构和移动原理十分简单，

但其作为人工智能技术在智能车领域的早期应用，具有里程碑式的意义，并为后续智能车的研究奠定了重要基础。Shakey 配备了摄像头传感器、三角测距仪和碰撞传感器，用于环境感知；通过驱动电机完成车辆控制任务，并利用人工智能技术进行环境分析与路径规划。随着硬件技术和人工智能技术的迅猛发展，如今的智能车和 Shakey 已经有了很大的不同，但其整体框架结构仍然具有一定的相似性。一个典型的智能车通常包括如图 1-2 示的几个模块：感知分析层、决策规划层及控制优化层。感知分析层通过多种传感器获取外部环境信息，并对数据进行解析和处理，类似于智能车的“眼睛”。常见的传感器包括鱼眼摄像头、深度相机、单线或多线激光雷达以及超声波雷达等，它们协同工作以提供全面的环境感知能力。决策规划层基于感知分析层和车辆提供的数据，评估环境状况并制定合理的行驶策略。该层负责规划一条安全、平稳且符合车辆运动学约束的最优行驶路径，为车辆的下一步行动提供决策支持。控制优化层则依据决策规划层生成的指令，精准执行车辆的运动控制，确保其按照规划的路径稳定运行，实现高效、安全的智能驾驶[11]。

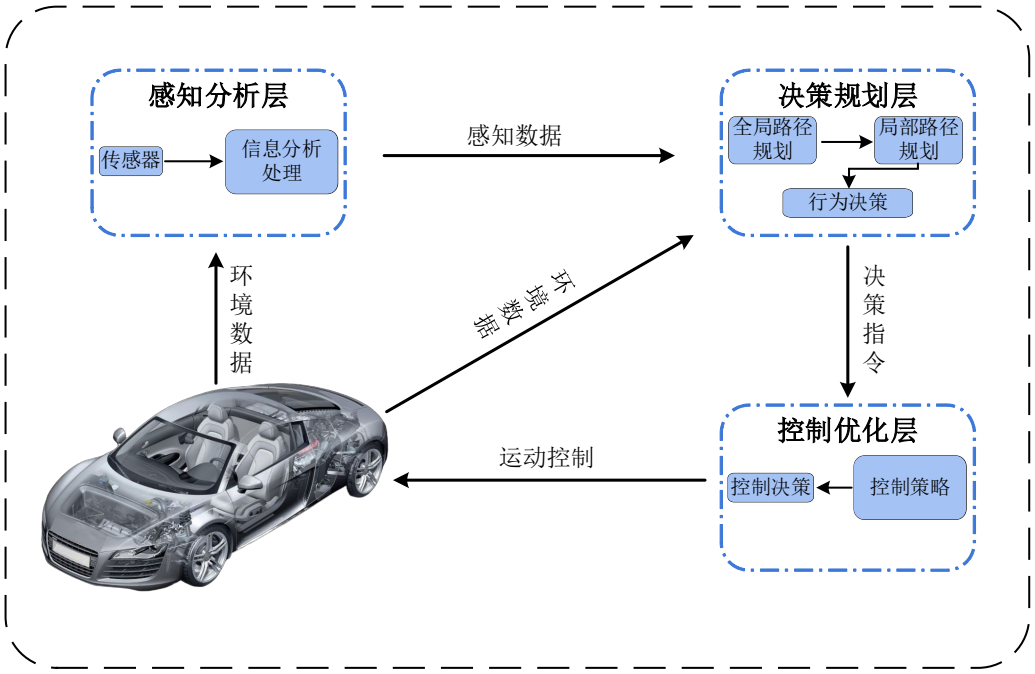


图 1-2 智能车架构

路径规划是智能车决策规划层的关键技术，对于智能车的导航、行驶、避障起着不可或缺的作用，也是实现底盘控制的前提条件。随着智能车行业应用场景

的变化，路径规划算法面临新的挑战。例如，在流水化制造车间中，智能车通常采用固定模式，并依赖全局路径规划算法完成路径规划。然而，该方法难以适应动态环境变化，且在与外界环境交互时可能引发不稳定因素。因此，在复杂动态环境下，如何确保路径规划的安全性与高效性，实时规避动态障碍物，已成为智能车自主导航亟待解决的关键技术问题[12-16]。

1.2 国内外研究现状分析

1.2.1 智能车研究现状

自 20 世纪 70 年代起，欧美国家的科研机构与大型科技公司便开启了智能车技术的研究探索。1984 年，美国国防高级研究计划署（DARPA）与陆军合作启动了具有里程碑意义的自主地面车辆（ALV）计划，该计划的实施标志着智能车研究正式进入系统化发展的新阶段。为进一步加速技术突破，美国于 2004-2007 年连续举办了三届 DARPA 无人驾驶挑战赛。该赛事引起了研究者对智能车的关注，使得大量的研究者投入到智能车技术的研发中。在 2007 年第三届赛事，卡内基梅隆大学的 Boss 团队以 4 小时 10 分的成绩成功完成全程并夺冠。历代智能车如图 1-3 所示。2010 年，谷歌公司研发的智能车已经实现了车道保持、自适应巡航和车队控制等功能；2015 年，百度研发的智能车成功实现最高时速可达 100km/h 的技术突破。次年，该款智能车获得美国车辆管理局的批准，允许其在特定路段进行智能车的道路测试。

(a) ALV　　(b) DARPA　　(c) Boss

图 1-3 历代智能车

在技术标准制定方面，美国高速公路安全管理局（NHTSA）于 2013 年首次提出了智能车分级体系[17]，为行业发展提供了重要参考框架。该体系将自动驾驶技术分为五个等级：L0 级为完全手动驾驶，驾驶员需要独立完成所有操作，

车辆仅具备基础的辅助驾驶功能；L1 级为驾驶辅助，车辆可协助完成自适应巡航、车道保持等简单操作；L2 级为部分自动驾驶，车辆可同时控制车速和方向，实现自动泊车、自动变道等功能，但驾驶员仍需保持对路况的持续监控；L3 级为条件自动驾驶，在特定环境（如高速公路）下可完全自主完成驾驶任务，驾驶员可暂时脱离驾驶操作；L4 级为高度自动驾驶，在限定区域内可实现完全自主驾驶；L5 级则为完全自动驾驶，无需任何人工干预。但受限于硬件性能、算法复杂度等多方面因素，当前主流的智能车仍处于 L2 级自动驾驶水平[18-19]。

虽然目前主流的智能车大都处于 L1 至 L2+的水平，尚未实现完全意义上的智能化，但是随着电子器件、感知技术、规划算法以及决策策略等技术的进步，L3 级乃至更高级别的智能车技术的实现已指日可待。

1.2.2 路径规划算法研究现状

自智能车的概念提出起，路径规划作为智能车驾驶的关键技术受到国内外研究学者的关注[20-22]。其核心目标是生成可行、安全且高效的轨迹，为车辆的控制决策提供基础。为了加快路径规划算法的发展，在过去的几十年里，研究学者们先后提出了 A*算法、RRT*算法等经典的路径规划算法。经典路径规划算法的时间线如图 1-4 所示。为应对复杂多变的环境，智能车的路径规划根据外界环境的不同，通常分为全局静态路径规划和局部动态路径规划两类。

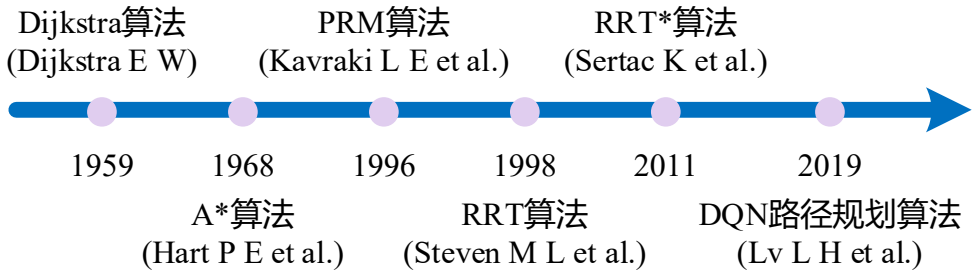


图 1-4　经典路径规划算法时间线图

全局静态路径规划算法主要分为基于栅格搜索的最短路径规划算法和基于采样的路径规划算法。A*算法[23]是广度优先遍历思想和启发式思想相融合的搜索算法，利用启发式信息引导路径的扩展方向，加快路径规划的收敛速度。但该算法的计算时间和路径代价会随着地图的扩大呈指数级增长。针对 A*算法生成路径代价高的问题，Huang 等[24]人提出一种基于 PSO 算法的 A*-PSO 算法，该

算法中超参数较多，对于不同环境难以选择一组具有泛用性的参数组合。针对A*算法计算时间长的问题，Ou 等人[25]提出一种基于自适应可见性图和边缘计算的混合路径规划方法，该方法融合了自适应可见性图与 A* 算法，以生成多条可行路径，并从中选择最优路径。为提高算法的收敛速度，作者将其分别部署于 CPU 和 GPU。然而，该算法在复杂环境下仍难以找到全局最优路径。

基于栅格搜索的路径规划算法存在收敛速度慢、路径代价高等缺点，而基于采样的路径规划算法利用采样的思想探索环境对路径进行规划，加快了规划的收敛速度、降低了路径代价。概率路线图(PRM)和快速扩展随机树(RRT)是经典的基于采样的路径规划算法。PRM 的优势在于其规划路径的过程与状态空间的维数无关，但是它在复杂环境或狭窄环境中收敛速度慢甚至不收敛。为了能找到目标点并且避开障碍物，RRT 算法通过在地图上进行随机采样扩展树的叶子节点以规划路径，但是这也导致规划的路径存在冗余点多、路径代价高等问题。2011 年 MIT 的 Sertac 和 Emilio 提出了 RRT*算法，设计了重选父节点和重新布线过程，改进了 RRT 算法的路径代价高的缺点。但是 RRT*算法仍采用随机采样方法对树进行扩展，导致复杂环境下算法收敛时间长、无效采样点多。针对 RRT*算收敛速度慢的问题，Qureshi 等人[26]将人工势场算法(Artificial Potential Field, APF)与 RRT*算法融合，提出了 PRRT*算法。PRRT*算法使用人工势场思想改进了扩展节点的策略，使随机树沿着势场降低的方向进行生长，以达到快速搜索目标点的目的。但是由于引入了人工势场，PRRT*算法在复杂环境中出现局部最优的问题。Fan 等人[27]针对 PRRT*算法的局部最优问题，提出一种基于目标偏向的双向路径规划算法。该算法使采样点以一定的概率导向目标点，在随机性的基础上增加目标导向性。但是算法只解决了简单环境下的局部最优问题，对于复杂环境或狭窄环境，仍存在局部最优问题。Zhang 等人[28]将海鞘群算法(Slap Swarm Algorithm, SSA)融入到 RRT*算法中。在 RRT*算法的随机采样过程引入海鞘群捕食机制和自适应领导者结构，减少了无效采样点的数量，提高了算法的全局寻优能力。但是由于 SSA 是种群优化算法对初始种群的配置依赖性较大，同时 SSA 算法迭代的次数多，导致算法收敛速度较慢。李宗刚等人[29]针对传统的深度 Q 网络(Deep Q Network, DQN)路径规划时存在的收敛速度慢、训练时间长等问题，

提出一种基于角度搜索的 DQN 路径规划算法。该算法通过确定搜索域，减少栅格点的搜索次数，进而加快算法收敛。张磊等人[30]针对深度双 Q 学习网络(DDQN)的路径规划算法在复杂环境下训练时间长、搜索有限等问题，提出一种改进的深度双 Q 网络学习算法，通过使用竞争网格结构对 DDQN 算法进行先验估计，加快策略的学习速度，增大搜索范围。

全局路径规划算法规划的是全局静态路径，无法完成动态避障任务。因此，动态环境下要使用局部路径规划算法以完成动态避障任务。DWA 算法通过对速度空间施加约束，以满足车辆的运动学模型要求。然后该算法在速度空间中搜索无人车的最优速度组合，预测局部路径以完成动态避障和车辆的规划任务，但 DWA 算法是局部路径规划算法容易陷入局部最优。针对 DWA 算法的这一问题，He 等人[31]设计了一种考虑时间因素的 DAA-star 算法，增强了算法的局部规划能力；Zhou 等人[32]提出一种 RRT*-FDWA 算法，解决了简单动态环境下的局部最优问题。 但以上两种方案在复杂动态环境下并不适用，并且这两种方案存在计算量大的问题，使算法收敛速度变慢。Kim 等人[33]提出了一种改进的 DWA 算法，考虑了障碍物的速度和方向。该算法将障碍物识别区域定义为椭圆，并在其中反映障碍物的速度。通过与现有 DWA 算法的比较，结果表明，改进的 DWA 算法能更安全地避开动态障碍物。Wu 等人[34]在 TEB 算法基础上，添加基于欧式距离的最短路径约束，解决了在复杂环境中规划路径时易出现的局部绕道问题。Wang 等人[35]针对多行为耦合时强化学习收敛慢的问题，提出了一种基于轨迹安全性和能耗优化的多行为批评家策略，有效提升了学习效率和避障的实时性。Yin 等人[36]结合深度强化学习与 LiDAR 技术，利用端到端网络预测动作的 Q 值，通过 n 步自举和深度 Q 网络架构实现了探索与利用的平衡，并以奖励值评估探索效果。徐建华[37]等人针对 AMR 在未知环境中轨迹规划过长的问题，引入距离梯度和角度梯度引导优化路径，提升了收敛速度。

综上所述，路径规划作为智能车的关键技术，直接影响其导航精度和安全性能。然而，目前国内外的路径规划算法在收敛速度方面仍有改进空间，局部路径规划的自主避障能力也较为薄弱。因此，深入研究智能车的路径规划算法具有重要意义。

1.3 本文的研究内容及章节安排

1.3.1 主要研究内容

本文主要研究智能车的全局路径规划算法和局部路径规划算法，通过两种算法的有机融合完成复杂环境下的路径规划任务。本文主要介绍了智能车路径规划的基础理论，通过提出的 PSRRT*算法规划全局路径提高规划效率，然后利用设计的 BDWAHT 算法进行局部路径规划，以提高路径的安全性和实时性，最后使用 ROS 平台进行仿真实验和实车实验验证算法的有效性和普适性。

1.3.2 论文章节安排

根据上述的研究内容，本文的章节架构如图 1-5 所示，具体安排如下：

第 1 章绪论。本章主要介绍智能车路径规划算法的发展过程和意义，并对国内外研究学者现阶段的研究内容进行分析介绍。

第 2 章智能车路径规划基础理论。本章主要对智能车的物理模型及环境地图的建模方法进行介绍，并对现阶段主流的全局路径规划算法和局部路径规划算法进行分析介绍。

第 3 章基于 PSRRT*算法的全局路径规划算法研究。本章对传统的 RRT*算法进行改进，设计精确采样模型、自适应节点扩展模型改善算法性能，并对全局路径进行平滑处理，增强全局路径的平滑性。为了验证算法的性能，本章设置了不同复杂度的测试环境下用于验证 PSRRT*算法的有效性。

第 4 章基于 BDWAHT 算法的局部路径规划算法研究。针对动态环境下传统局部路径规划算法计算复杂度高、局部最优等问题，本章提出一种 BDWAHT 算法。该算法通过设计双动态窗口模型减少计算复杂度，并基于热传递方程动态调节窗口大小。最后将提出的 PSRRT*算法和 BDWAHT 算法进行融合，以完成复杂环境下的路径规划任务。通过仿真结果分析，验证了 BDWAHT 算法的性能。

第 5 章基于 ROS 平台的智能车实验测试。本章对 ROS 平台进行了简要介绍，搭建了 ROS 仿真测试平台，构建了复杂动态测试环境并对 PSRRT*-BDWAHT 融合算法进行了可行性和普适性的验证测试。

第 6 章总结与展望。本章对本文的主要研究进行了总结分析，并针对现阶段

智能车的发展进行分析，明确了下一步的研究方向。

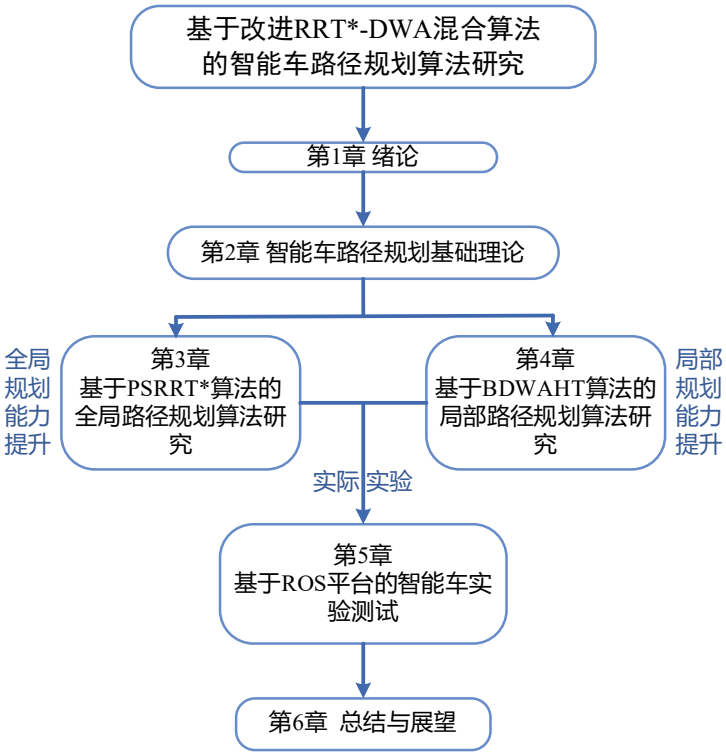


图 1-5 本文章节内容架构

第2章 智能车路径规划算法基础理论

路径规划算法的目标是规划出一条无碰撞、安全且行驶效率高的路径。规划的路径通常使用收敛性和最优性来衡量其性能，以此判断路径规划算法的有效性。收敛性是指在给点路径起点和终点的情况下，算法能否收敛到一个可行解。最优性则是指规划的路径在路径长度、平滑性、安全性和规划时间等方面是否达到最优[38]。本章将对汽车运动学建模、环境地图建模，全局路径规划和局部路径规划进行相关介绍。

2.1 车辆运动学建模

数学模型是针对参照某种事物系统的特征或数量依存关系，使用数学语言，概括地或近似地表述出的一种数学结构，这种数学结构是借助于数学符号描绘某种系统的纯关系结构。作为一种解决问题和思考的有效方法，数学模型能够为处理实际问题提供科学而合理的决策依据[39-40]。基于智能车物理运动约束的考虑，本研究选取以后轴为中心的阿克曼车辆模型作为研究对象。

以后轴为中心的车辆模型将参考系坐标固定在车辆的后轴中心，可以将车辆运动分解为纵向平移和绕后轴旋转的两个分量。这种几何关系满足车辆的非完整约束条件，即车辆横向速度分量在后轴中心处为零。并且当车辆速度较慢时，轮胎侧偏角对整个运动轨迹的影响较小可以忽略，此时该模型能准确的预测车辆的运动轨迹。车辆运动模型如图 2-1 所示，并做出以下假设：

（1）假定车辆的运动仅限于水平面上，不考虑垂直方向上的任何运动分量，即将车辆视为二维平面上的运动体。

（2）假设车辆在前进过程中速度变化平缓，即忽略加速或减速引起的前后轴载荷转移效应。

（3）假设车辆为刚体模型，即车辆在运动过程中不发生形变，其结构和几何形状保持不变。

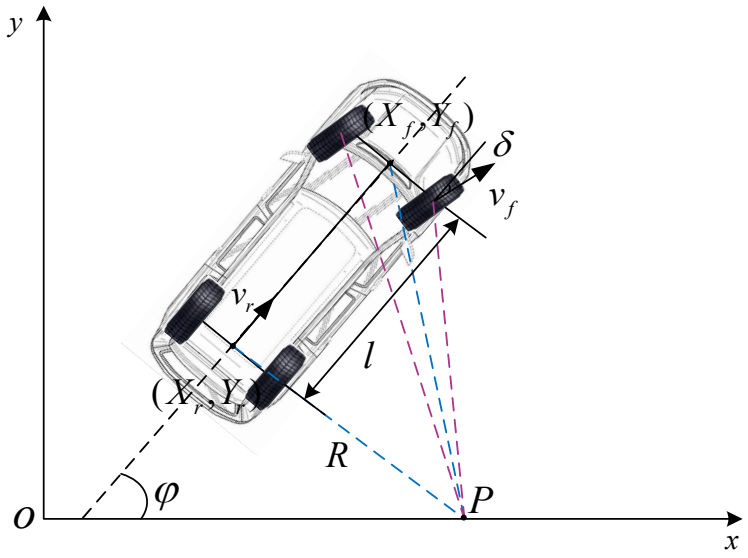


图 2-1 车辆运动学模型

图 2-1 中 (X_r,Y_r) 为车辆后轴中心坐标，(X_f,Y_f) 为车辆前轴中心坐标。δ 表示车辆前轮转角，φ 表示车辆航向相对于 X 轴的偏转角度。l 表示车辆轴距，P 为瞬时滚动中心，R 表示转弯半径。

车辆运动学模型如(2-1)所示：

$$\begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} cos\varphi \\ sin\varphi \\ \dfrac{tan\delta}{l} \end{bmatrix} v_r \tag{2-1}$$

2.2 环境地图建模

环境地图建模的核心在于将复杂且动态的真实世界环境信息提取为计算机能够理解和处理的形式。这一过程不仅涉及到对环境特征的提取，还包括如何以一种结构化的方式表示这些特征，以便路径规划算法能够高效利用这些信息。环境建模是实现路径规划的基础，尤其在智能车这一领域，一个准确的环境模型能够显著提升路径规划的准确性和效率。在构建环境地图模型时，必须考虑以下几个关键因素：

（1）可表征性：即模型是否能够充分表征出环境中的所有关键要素，例如道路、行人、车辆、障碍物等。

（2）可扩展性：新环境的模型能否能够方便地集成到现有的模型中，以便在动态变化的环境中进行实时更新。

（3）可解释性：模型应具备良好的数学表征形式，确保路径规划算法能够高效解析并执行实时计算，特别是在动态复杂环境下的快速决策场景中。

在环境地图建模中，常见的方法包括拓扑图法、栅格地图法。本节将对这两种建模方法的优缺点进行分析和介绍。

2.2.1 拓扑图法

作为环境建模领域的经典方法，拓扑图法[41]在智能车导航系统中占据重要地位。该方法通过建立节点-边网络模型实现环境的抽象化表达，其中几何空间中的道路交叉点、建筑特征点等关键位置被定义为节点，节点间的通行路径则量化为连接边。由于该方法仅存储节点和连接边的信息，极大的减少了数据存储空间，使得该方法在大尺寸环境下有着更好的表现。然而，拓扑图的简化表达忽略了环境的某些细节信息，导致在进行精细化的路径规划时表现较差。图 2-2 展示了拓扑图法的结构特征，其中圆形符号 P_i 表示节点，蓝色实线表示节点间的有效连接关系。

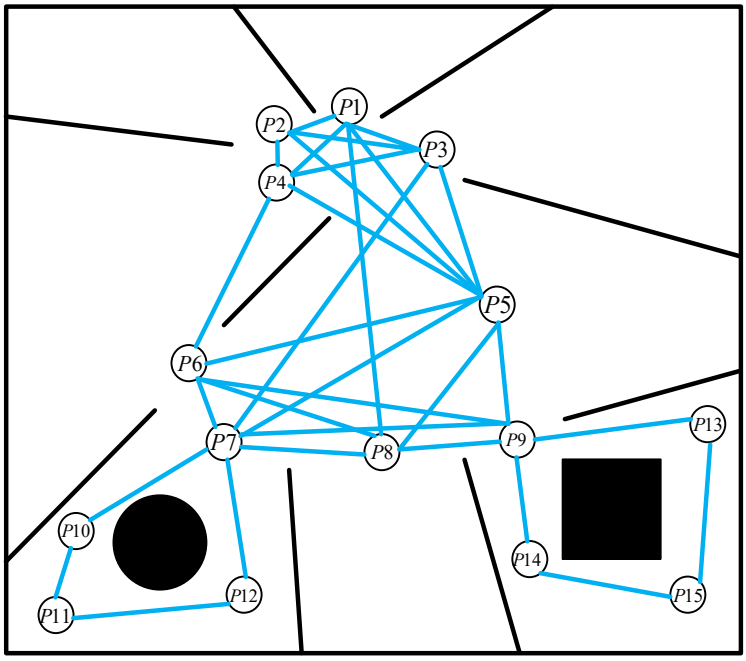


图 2-2 拓扑地图

2.2.2 栅格地图法

栅格地图法[42]是环境地图建模方法中最为常见的一种。该方法通过构建二维数组来表示环境网格的占据信息，首先将整个地图进行栅格化，并对每个栅格进行检测赋予状态信息。通常使用数字“1”表示该栅格已经被占用，数字“0”表示该栅格处于空闲状态，可以通行。在栅格地图中，每个栅格都有唯一的状态信息，能够简单直观地了解周围环境。同时，栅格地图将路径规划问题转化为一个求解多约束的目标单元格规划问题。栅格地图如图 2-3 所示，黑色部分表示占据的栅格，白色部分表示空闲状态的栅格。

栅格地图的构建方式简单，能够直观地反映环境的复杂性，同时算法也更容易理解环境信息，从而有助于路径规划算法的实施。在地图规模较小的环境中，该方法可以清晰地表达环境特征，构建的地图模型具有较强的鲁棒性，在一定程度上能够减少计算的复杂度。但随着地图规模的扩大，栅格地图法的复杂度呈指数级增长，因此该方法不适用于复杂的大尺寸环境。同时复杂的大尺寸环境中的障碍物多是不规则的，由栅格地图法构建的地图模型可能会丢失障碍物的细节信息，在某些需要精细化规划的场景下并不适用。

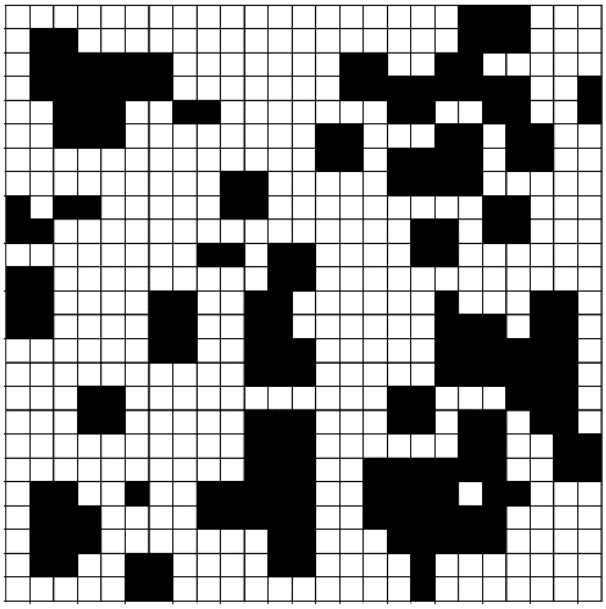


图 2-3 栅格地图

2.3 全局路径规划算法

全局路径规划算法主要应用于环境先验信息完全已知的场景，包括障碍物分布、环境边界约束及自由空间等结构化信息。该类算法基于完备的环境模型，通过求解最优化问题生成全局最优或次优路径，为智能车的导航提供全局参考路径。从外界环境变化的角度来看，全局路径规划算法属于静态路径规划算法。由于其需要完整的环境信息才能规划出有效的路径，这使得在动态环境或包含未知信息的复杂环境中，全局路径规划难以及时调整并规划有效的路径。因此，全局路径规划一般用于静态或封闭环境中。本节将全局路径规划算法分为传统路径规划算法和基于神经网络的路径规划算法，传统路径规划算法主要包括基于搜索的路径规划算法(如 Dijkstra 算法、A*算法等)、基于采样的路径规划算法(如 PRM 算法、RRT*算法等)和智能种群算法(如粒子群算法、蚁群算法等)。

2.3.1 基于搜索的路径规划算法

基于搜索的路径规划算法，也称为基于图搜索的路径规划算法，其本质思想是对外界环境进行简化，从而使用图搜索方法进行路径规划。在进行图搜索之前，需要对外界环境进行建模。由于栅格地图法与图搜索方法高度契合，因此在基于搜索的路径规划算法中通常使用栅格地图法进行环境建模。广度优先遍历思想和启发式搜索思想是基于搜索的路径规划算法中常用的两种思想，其中 Dijkstra 算法是广度优先遍历思想的经典代表，而 A*算法则是启发式搜索思想的经典代表。

Dijkstra 算法[43-45]从起始节点出发，首先遍历并计算其邻域节点到起始节点的代价值；随后迭代访问各节点的邻域节点，持续更新节点代价值，并以波前形式向外扩展搜索区域；当检测到目标节点或完成整个栅格地图的遍历时，算法终止；最终通过反向追踪机制提取从起始节点到目标节点的最优路径。Dijkstra 算法路径规划示意图如图 2-4 所示。由于 Dijkstra 算法采用的是广度优先遍历思想，只要栅格地图足够精细，如果存在全局最优路径，理论上就一定能够找到这条路径。然而，Dijkstra 算法的暴力搜索策略导致算法的计算资源消耗较大，收敛速度较慢。此外，栅格地图法在复杂环境下进行环境建模时，栅格地图法本身也需要较高的计算资源，这进一步增加了 Dijkstra 算法的计算资源需求。

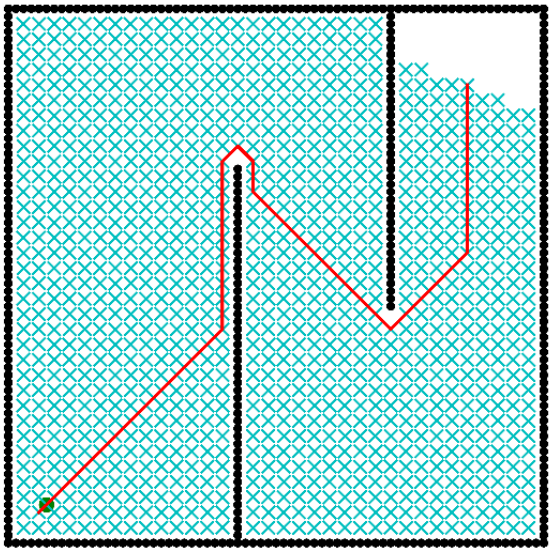


图 2-4 Dijkstra 算法路径规划示意图

针对 Dijkstra 算法计算复杂度高和收敛速度慢等问题，Peter Hart 于 1968 年提出了 A*算法。A*算法[46-48]自问世就被认为是静态环境中求解最短路径的最佳方法之一。该算法在 Dijkstra 算法的广度优先遍历思想上融合了启发式搜索思想，使用启发式信息来优化路径扩展方向，减少探索次数，从而加快算法的收敛速度。A*算法优先探索距离路径起点较近的节点，然后再根据启发式信息探索其他节点。尽管 A*算法相较于 Dijkstra 算法在计算复杂度、收敛速度等方面有所提升，但是由于图搜索的思想导致该算法在复杂环境下仍保持着一个较高的计算复杂度。同时由于使用的栅格地图法在复杂环境下也需要较高的计算资源，进一步加大了计算复杂度。A*算法路径规划示意图如图 2-5 所示。

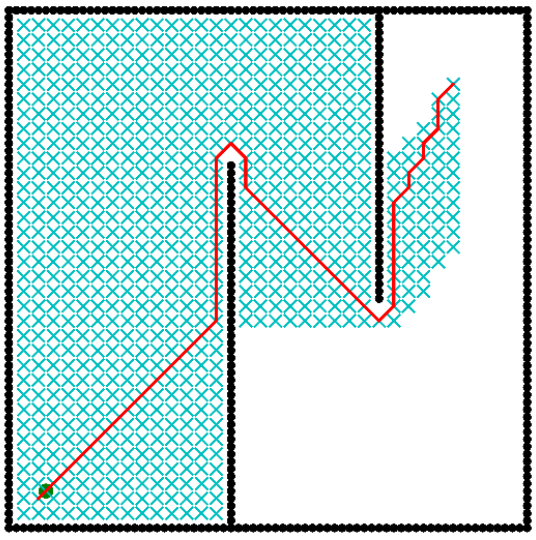


图 2-5 A*算法路径规划示意图

2.3.2 基于采样的路径规划算法

基于采样的路径规划算法利用采样思想探索环境对路径进行规划，加快了规划的收敛速度、降低了计算复杂度。同时由于其能很好地解决高维空间的路径规划问题，因此受到研究者的青睐。概率路线图算法(PRM)[49-51]，首先在地图上随机采样自由空间中的点，然后将这些点进行连接，最后从连接的线段中找出一条从起点到终点的无碰撞路径。PRM 的优势在于其规划路径的过程与状态空间的维数无关，但是它在复杂环境或狭窄环境中收敛速度慢甚至不收敛。PRM 算法规划示意图如图 2-6 所示。

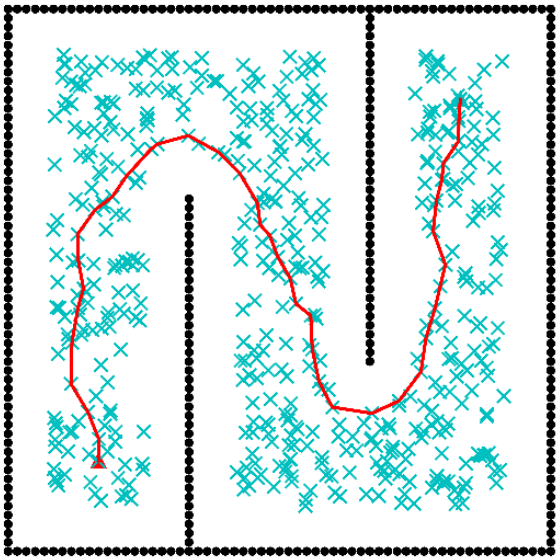


图 2-6 PRM 算法规划示意图

RRT*算法[52-53]通过随机采样来获得节点的候选点，然后通过判断采样点与随机树中最近节点间是否安全，进而扩展随机树。随后使用重新选择父节点策略优化路径长度，最后对随机树中的节点进行重新布线优化路径性能。为了能找到目标点并且避开障碍物，该算法通过在地图上进行随机采样扩展随机树以规划路径，但是这也导致规划的路径存在冗余点多、路径代价高等问题。同时复杂环境或狭窄环境中，由于环境复杂度高，算法需要在自由空间中重复采样、反复计算才能获取有效的采样点，这一过程加大了计算量。RRT*算法的规划示意图如图 2-7 所示。

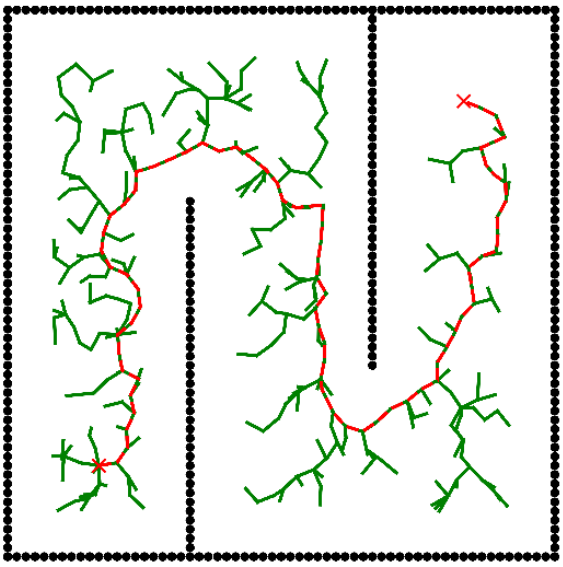


图 2-7 RRT*算法规划示意图

2.3.3 基于智能种群的路径规划算法

粒子群算法[54-55]是一种种群优化算法，模拟了鸟群在自然界中觅食的行为。该算法首先由个体自由搜索，由于种群个体的差异导致搜索效果不同，然后检测个体搜索结果，使所有个体向最优个体靠近，通过不断调整最优个体的位置和速度，以逐步搜索到最优解。为了解决路径规划问题，该算法将每个粒子的历史位置视为一条可能的路径。通过调整粒子的位置和速度，在解空间中寻找最优路径。粒子群算法的流程图如图 2-8 所示。

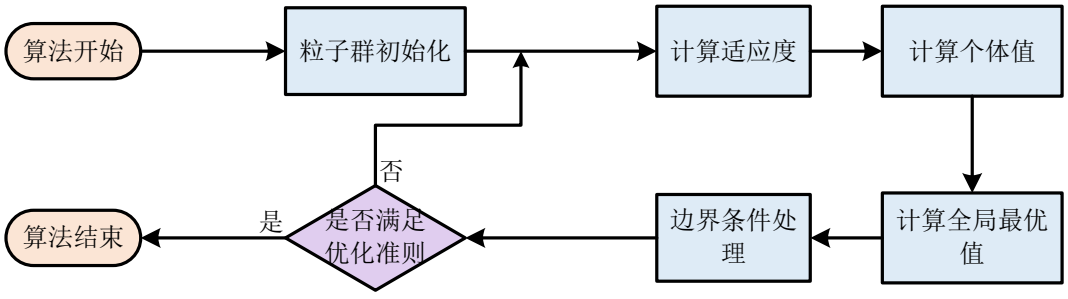


图 2-8 粒子群算法流程图[54]

蚁群算法[56-57]是一种基于蚂蚁觅食行为的概率算法，具有分布式计算、信息反馈和启发式搜索的特点，属于种群算法中的一种启发式全局优化方法。其基本原理是蚂蚁在寻找食物时释放信息素，从而影响其他蚂蚁的路径选择。信息素浓度越高，后续蚂蚁选择相同路径的可能性就越大，直至找到目标点。信息素浓度

会随着时间的推移和路径长度的增加而减少，因此最短路径上通常会积累更多信息素，从而吸引更多蚂蚁，形成正反馈机制。这种机制推动系统朝向最优解演化，使蚂蚁更快地找到起始点与目标点之间的最佳路径。蚁群算法的流程如图 2-9 所示。

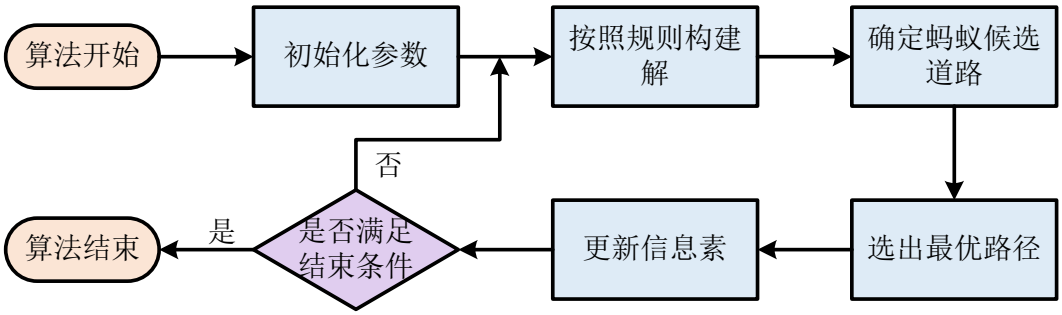


图 2-9 蚁群算法流程图[58]

2.3.4 基于神经网络的路径规划算法

随着人工智能技术的快速发展与广泛应用，基于神经网络的路径规划方法已成为研究热点。在众多神经网络架构中，生成对抗网络(Generative Adversarial Networks， GAN)与深度 Q 学习网络(Deep Q-Learning Network， DQN)因其独特的优势，在路径规划领域获得了广泛关注与应用。

基于 GAN 的路径规划算法[58-59]，通常先使用 GAN 网络在地图上生成包含可行路径的感兴趣区域，随后再使用基于采样或基于搜索的路径规划算法来寻找最优路径。基于 DQN 的路径规划算法[60-61]使用神经网络构建一个 Q 表，并根据 Q 表中每个单元的代价进行迭代优化。然而，当面对动态障碍物时，DQN 网络表现出一定的局限性。这主要是因为 DQN 网络在处理高速动态环境信息时，尤其是突发的、非线性的障碍物变化时，难以快速有效地更新其决策策略。这导致路径规划效率较低，甚至在某些情况下无法找到可行路径。

基于神经网络的路径规划算法存在以下几个缺点：

（1）数据需求量大。神经网络的训练通常需要大量的数据，尤其是在环境复杂的情况下，需要收集大量的环境样本进行训练。收集并标注这些数据的成本较高，并且在动态环境中需要不断更新训练数据。

（2）泛化能力有限。神经网络模型容易在训练集上表现良好，但在未出现

过的场景或环境中可能表现不佳。这是因为神经网络可能会对训练数据过拟合，导致在不同的环境或地图上无法有效应对环境变化。

（3）难以满足路径规划的实时性。路径规划通常要求快速的响应和实时计算，尤其是在动态或不断变化的环境中。然而，神经网络在处理复杂的规划问题上可能需要较长的计算时间。此外，推理过程受到硬件条件约束，可能导致实时性不足的问题。

2.4 局部路径规划算法

局部路径规划算法主要用于智能车的局部动态规划，专注于动态环境，具备较高的灵活性和实时性，属于在线路径规划算法。在全局路径规划算法的引导下，局部路径规划算法通过传感器实时采集和处理外界环境数据，以完成动态环境下的避障任务和路径规划任务。与全局路径规划算法相比，由于动态环境的多变性和不确定性，局部路径规划算法所规划的路径往往是局部最优的，甚至是不可达的。本节将重点介绍 DWA 算法、时间弹性带算法和模型预测控制算法。

2.4.1 DWA 算法

DWA 算法[62]由 Dieter Fox 等人在 1997 年提出的，主要用于智能车的局部路径规划和动态避障。该算法首先基于智能车的当前位置和速度状态，在速度空间中确定一个符合车辆运动学约束的采样速度范围。随后，算法会模拟车辆在不同速度条件下的轨迹，并通过评价函数对这些轨迹进行评分，选择评分最高的轨迹所对应的速度，作为车辆的下一步控制指令，使智能车按照最优轨迹运动。DWA 算法的流程图如图 2-10 所示。

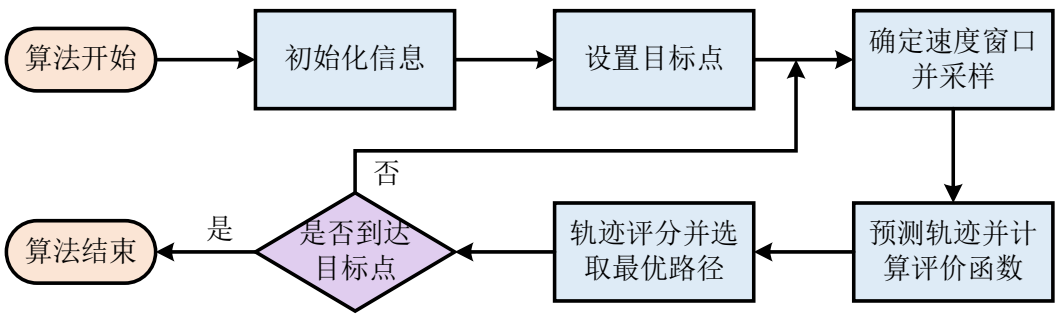


图 2-10 DWA 算法流程图[62]

在智能车局部路径规划任务中，DWA 算法将车辆的运动控制问题转化为线速度和角速度的控制问题，从而将避障问题转变为空间中的运动约束问题。算法通过引入车辆运动学约束条件，在规划域内求解最优运动轨迹，从而实现动态环境下的实时避障与路径规划。

2.4.2 时间弹性带算法

时间弹性带算法(Time Elastic Band，TEB)[63]是使用动态约束优化进行局部路径规划的方，它对全局参考路径附加一个时变约束并使其变形得到一条新的路径，以此来达到路径跟踪及避开静态或动态障碍的目的。TEB 算法对于智能车来说有较大的适用范围，在一些要求快速反应、外界环境变化剧烈的情形下可以很好的发挥作用。算法的基本思想是先构造出从起点到终点的一条初始局部路径，然后在这条路线上按一定间距选取 N 个控制点，再定义控制点间的时间间隔，进而建立约束条件。最后综合车辆运动学约束条件、汽车运动模型、障碍物约束以及路径约束等条件求解最优路径。

TEB 算法将智能车的当前状态和目标状态作为输入，综合考虑各项约束，经过求解超图最终生成带有时间信息的平滑轨迹。该算法的适应性较强，适用于多种常见车辆模型（如差动驱动模型、阿克曼转向模型等），并且具备一定的前瞻性，能够提前应对潜在的动态障碍物。尽管 TEB 算法具有较好的规划效果，但其计算复杂度较高，容易导致速度波动大和控制响应不稳定等问题。在实际应用中，这些缺点可能会影响路径的平滑性和控制精度，因此通常需要使用其他优化算法来改善 TEB 算法的实时性和稳定性。

2.4.3 模型预测控制算法

模型预测控制算法[64]是一种先进的优化控制算法，其核心思想是基于当前状态预测未来一段时间内的最优轨迹，并计算相应的控制输入，以确保车辆能够平稳地沿着该轨迹行驶。其主要流程包括三个步骤：预测模型、滚动优化和反馈校正。模型预测控制算法流程图如图 2-11 所示。

首先，算法根据车辆运动学模型构建未来轨迹的预测模型，预测模型使用历史数据和当前车辆的输入来预测未来车辆的速度控制输出，输出控制着车辆沿着

全局路径进行移动。但是，由于外界环境的变化、数据噪声、模型偏差等因素，导致预测输出与实际输出之间出存在误差。此时，模型预测控制算法会采用滚动优化策略减小误差，即在每个时间步长内只执行下一个时刻的预测输出，舍弃其他时刻的预测结果，不断地调整输出以求解最优路径。这种优化方法使得该算法在动态环境中能快速的更新预测轨迹，增强算法的实时性。同时该算法使用实际测量值和预测输出进行实时校正弥补因建立的预测模型不准导致的误差，进一步增强了局部路径的精度和可靠性。

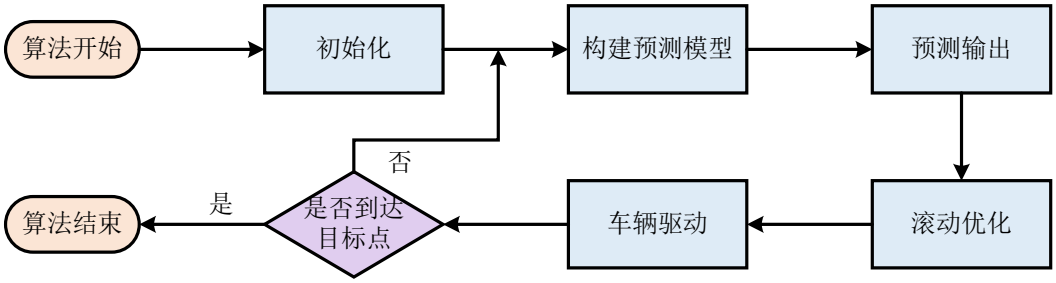


图 2-11 模型预测控制算法流程图[64]

2.5 本章小结

本章首先介绍了汽车的运动学建模方法，然后对主流的环境地图建模方法进行了介绍，主要分析拓扑图法和栅格地图法的特点及其应用场景；最后分别介绍了主流的全局路径规划算法和局部路径规划算法的优缺点、差异性以及适用场景。

第3章 基于 PSRRT*算法的全局路径规划算法研究

针对传统全局路径规划算法存在计算复杂度大、收敛速度慢、路径平滑性低等问题，本章提出一种 PSRRT*算法。首先，建立精确采样模型获得路径采样点，以提高采样的导向性并减少无效采样点的数量，从而加快算法的收敛速度；其次，建立自适应节点扩展模型，该模型基于采样点的分布特征动态调整路径扩展方向，通过构建新型引力场函数及自适应优化斥力场参数，在保障路径规划质量的同时显著降低计算复杂度，从而生成初始的全局路径；最后，对初始的全局路径进行剪枝和平滑处理，得到全局最优路径。为验证算法性能，本章设计了多组对比实验：选取 500×500 cm 和 1000×1000 cm 两种地图尺度，并在每种尺度下设置 20%、35%、40%三种障碍物密度等级，对 PSRRT*算法进行系统性仿真测试。实验结果表明，PSRRT*算法在全局路径规划任务中展现出显著的性能优势，在求解效率与计算资源方面均有明显提升。

3.1 RRT*算法原理

RRT*算法[65-66]属于 RRT 算法的优化算法，因其无需对环境进行复杂建模和良好的规划性能受到研究者的青睐。该算法将规划起点作为根节点构建随机树，通过不断生长树来探索环境并寻找最优路径。在树的生长过程中主要包含以下几个过程：

（1）在工作空间中进行随机均匀采样，获取候选采样点，并从已有树中选择与其距离最近的节点作为扩展基点;

（2）基于矢量方向从基点向采样点生成新的节点，通过碰撞检测确认该扩展路径在环境中无障碍后，将新节点添加至树中；

（3） 在新节点附近搜索邻域节点，选取使路径代价最小的节点作为其父节点，实现路径代价优化；

（4）在新节点邻域内查找其他节点，若经由新节点可使其路径成本降低且满足无碰撞要求，则更新其父节点以完成树的局部重构，提升全局最优性。

RRT*算法通过不断地生长来探索外界环境，以找到全局最优路径，算法伪代码如算法 1 所示。

其中 $G=(V,E)$ 为最终的生成的随机树，V_1 为以起点为根节点的子随机树。伪代码中的 *RandomSamplie*(•) 为随机采样模型，原理如式(3-1)所示。

算法 1 快速扩展随机树算法 $(RRT*)$

Input: X_{obs},X_{start},X_{goal},$V=\phi$,E=ϕ,$step$

Output: G

```
1: V1.inited(X_start)
2: for m = 1 → M do
3:     q_rand = RandomSamplie(·)
4:     q_near = GetNeareastNode(q_rand1, V)
5:     q_new = Expend(q_rand, q_near, step)
6:     if CollisoinFree(q_near, q_new) then
7:         Q_near=Near(V,q_new)
8:         q_parent = ChooseParent(Q_near, q_new)
9:         V1.append(q_new)
10:        E.append(q_new, q_parent1)
11:        E→Rewire(V1,q_new1, Q_near)
12:    end if
13:    if X_goal  in  V then
14:        V = V1
15:    end if
16: end for
17: return G = (V, E)
```

$$R(X_{free},V)=\begin{cases}(x_{goal},y_{goal}) & ,r_R<0.1\\(x_{rand},y_{rand}) & ,r_R\geq 0.1\end{cases} \tag{3-1}$$

其中 r_R 为[0,1]间的随机数，(x_{goal},y_{goal}) 为终点坐标，(x_{rand},y_{rand}) 为随机点坐标。

算法获得采样点后根据设定的扩展步长 *step* 、随机树上的最近节点 q_{near} 和采样点 q_{rand} 的信息生成新的叶子节点 q_{new}，从而扩展随机树。

$$q_{new}=q_{near}+step*\frac{\overrightarrow{q_{rand}q_{near}}}{\left\|\overrightarrow{q_{rand}q_{near}}\right\|} \tag{3-2}$$

重新选父节点和重新布线过程主要是为了优化随机树的结构，减少路径代价，如伪代码 8-12 行所示。

最后算法通过检测目标节点是否被搜索来判断是否完成规划。虽然 RRT*算法采用重新选父节点和重新布线提升了算法的规划性能，但是其使用的随机采样模型和固定的扩展方式，导致该算法在复杂环境下难以快速找到全局最优解。

3.2 PSRRT*算法

针对 RRT*算法在复杂环境下收敛速度慢、难以找到全局最优解的问题，本节提出一种 PSRRT*算法。首先，在采样阶段提出一种精确采样模型，该模型通过引入目标导向的启发式信息优化采样过程，显著提高了有效采样点的比例，同时降低了无效采样点的数量，从而提升算法的收敛速度。其次，在路径扩展阶段，提出基于向量扩展和新型人工势场扩展的自适应节点扩展模型，该模型不仅有效降低了算法的计算复杂度，而且克服了传统人工势场在复杂环境下的局部最优问题。最后，在路径优化阶段，通过引入剪枝策略和 B 样条平滑机制改进路径的代价和平滑性。PSRRT*算法架构如图 3-1 所示。

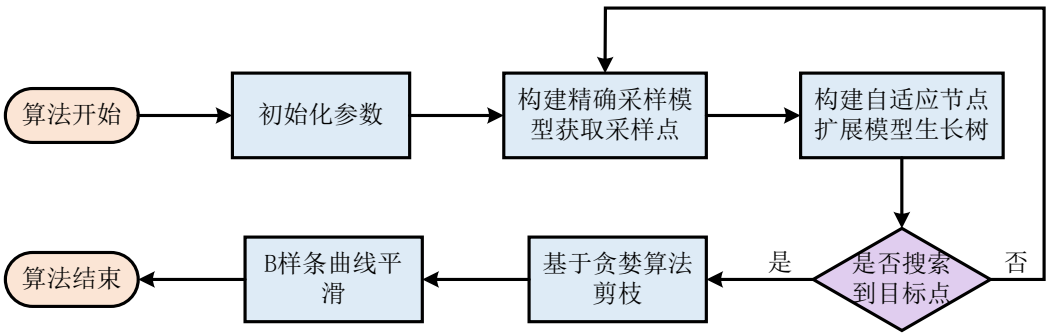


图 3-1 PSRRT*算法架构图

3.2.1 精确采样模型

RRT*算法的采样阶段使用的是随机采样模型，虽然能够找到较好的路径，但由于采样不具备导向性，导致冗余点过多，算法收敛速度慢。为了解决这个问题，本节建立了一种精确采样模型。

首先，引入一个采样阈值用于切换启发式圆域采样模型和随机采样模型如式(3-3)所示。当执行随机采样时，采用传统的偏置采样模型获取随机点；当使用启发式圆域采样时，算法首先在设定的自适应圆域内采样多个候选采用点，然后通过评估启发式信息选择最优候选点作为最终的采样点。

$$q_{rand}=\begin{cases}H(X_{free},q_{near},V) & ,r\geq\varsigma\\ R(X_{free},V) & ,r<\varsigma\end{cases} \tag{3-3}$$

其中 q_{rand} 为采样点，r 为[0,1]间的随机数，用于判断使用的采样模型，ς 为采样

阈值，X_{free} 为自由空间，V 为随机树，q_{near} 为最近节点，$H(\bullet)$ 为启发式圆域采样模型，$R(\bullet)$ 为偏置采样模型。

偏置采样模型描述为：

$$R(X_{free},V)=\begin{cases}(x_{goal},y_{goal}) & ,r_R<0.1\\(x_{rand},y_{rand}) & ,r_R\geq 0.1\end{cases} \tag{3-4}$$

式(3-4)中 r_R 为[0,1]间的随机数，(x_{goal},y_{goal}) 为终点坐标，(x_{rand},y_{rand}) 为随机点坐标。

启发式圆域采样模型的核心在于自适应采样圆域的动态确定机制，该机制通过实时评估环境特征和已搜索的节点数量来优化采样范围，从而在探索效率和采样精度之间实现最佳平衡。

设置自适应圆域 C 为：

$$C=\begin{cases}(c=(x_q,y_q),R=D*2^{\frac{125-n}{250}}) & ,n\leq 125\\(c=(x_q,y_q),R=D*\frac{1}{5}(1+3e^{\frac{125-n}{125}})) & ,n>125\end{cases} \tag{3-5}$$

式(3-5)中 c 为自适应圆域的圆心坐标，R 为自适应圆域的半径，(x_q,y_q) 为当前节点坐标，D 为当前节点到终点的欧式距离，n 为随机树上的节点数。

在路径扩展阶段，为提升采样质量与方向选择的合理性，本节在已确定的自适应圆形采样域内，融合环境信息对自由空间进行采样，生成候选点集合。该采样策略不仅确保了探索区域的覆盖性，同时有效规避了障碍物聚集区域，从而提升了采样点的可行性与多样性。为从候选点集合中筛选最优扩展方向，进一步引导树结构朝向代价更低的路径延展，构建启发式代价函数 $f(\bullet)$，从候选点集合中选择代价函数最小的点作为采样点。

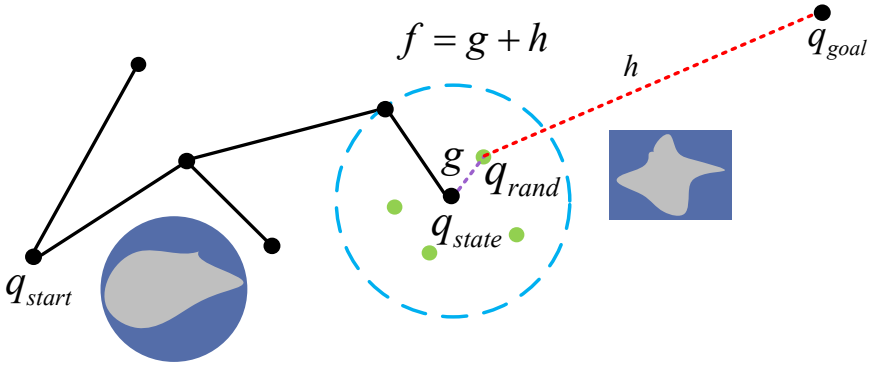


图 3-2 启发式采样图

启发式采样过程如图 3-2 所示，其启发式代价 f 可描述为：

$$f(q^*)=g(q^*)+h(q^*) \tag{3-6}$$

$$g(q^*)=g(q_{near})+M(q_{near},q^*) \tag{3-7}$$

$$h(q^*)=M(q^*,q_{goal}) \tag{3-8}$$

其中 q^* 为候选点，q_{goal} 为终点，$g(q^*)$ 为累积代价，$h(q^*)$ 为预估代价，$M(\bullet)$ 为曼哈顿距离。在计算启发式代价时，之所以选择使用曼哈顿距离来计算代价，主要曼哈顿距离避免了耗时的平方和开方运算，其简单的绝对值相加特性将计算复杂度从 $O(n^2)$ 降低至 $O(n)$，显著降低了算法的计算复杂度。

综上，启发式圆域采样模型为：

$$H(X_{free},q_{near},V)=\arg\min_{q^*\in Q}(f(q^*)) \tag{3-9}$$

$$Q=R(C) \tag{3-10}$$

其中 q^* 为候选点，Q 为候选点集合，$R(\bullet)$ 为偏置采样函数，C 为自适应圆域。

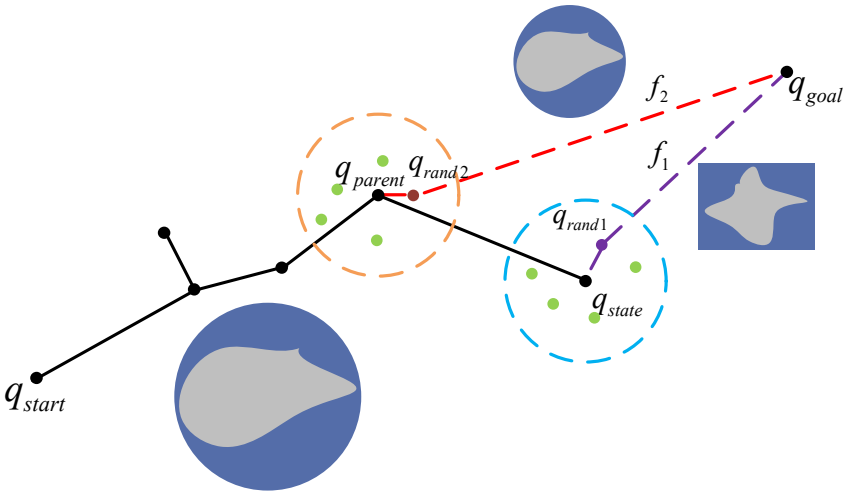


图 3-3 精确采样图

在复杂环境中，由于障碍物密集、空间受限等因素的干扰，当前扩展节点可能处于次优甚至无效的采样区域内，导致其生成的采样点难以引导随机树朝向更优路径扩展，此时需要引入父节点信息来优化采样策略。如图 3-3 所示，当前节点 q_{state} 所在的采样域中生成的候选采样点 q_{rand1} 难以满足路径质量或规划效率要求，而其父节点 q_{parent} 所在区域则能够提供更加优质的采样点 q_{rand2}。因此，通过代价函数选择最优采样点，从而确保节点扩展过程能够更有效地避障并朝向全局

最优路径发展。精确采样模型过程如算法 2 所示。

算法 2 精确采样模型

Input: $q_{near}, q_{goal}, X_{free}, V$

Output: q_{rand}

if $random < \xi$ **then**
  if $randomR < 0.1$ **then**
    $q_{rand} = (x_{goal}, y_{goal})$
  else
    $q_{rand} = Sample(X_{free})$
  end if
else
  $D = Dist(q_{near}, x_{goal})$
  if $n < 125$ **then**
    $R = D * 2^{\frac{125-n}{250}}$
  else
    $R = D * \frac{1}{5}(1 + 3e^{\frac{125-n}{125}})$
  end if
end if
$O = (q_{near}.x, q_{near}.y)$
$q_{rand1} = HCS(O, R)$
$q_{rand2} = HCS(O.parent, R)$
$q_{rand} = ChooseOpt(q_{rand1}, q_{rand2})$
return q_{rand}

其中 $HCS(\bullet)$ 函数获取精确采样点，q_{rand1} 和 q_{rand2} 分别为当前节点和其父节点的精确采样点，$ChooseOpt(\bullet)$ 函数从 q_{rand1} 和 q_{rand2} 中选择目标导向性更强的采样点。

3.2.2 自适应节点扩展模型

在基于采样的路径规划算法中，路径的扩展策略直接影响着算法的收敛效率与路径质量。目前主流的扩展方式主要包括向量扩展法与人工势场扩展法两种方法。向量扩展法是最常用的扩展策略之一，其核心思想是以当前树节点与采样点之间的方向向量作为路径扩展的方向，并据此生成新的候选节点。该方法高度依赖采样点的位置，往往无法充分利用环境中的目标信息，导致树结构可能向非最优方向扩展，增加了搜索冗余和计算负担。人工势场扩展法则是一种基于虚拟力场理论的路径引导策略，广泛应用于智能车路径规划任务中。该方法通过构建一个由目标点产生吸引力、障碍物产生斥力的人工势场模型，将车辆视作在势场中

运动的质点，受力引导其朝向目标区域并避开障碍物。传统人工势场的原理如图3-4所示，终点q_{goal}对车辆q_{near}产生引力F_{att}，障碍物X_{obs}对车辆产生斥力F_{rep}，D_0为斥力作用范围，合力F引导车辆向终点移动。

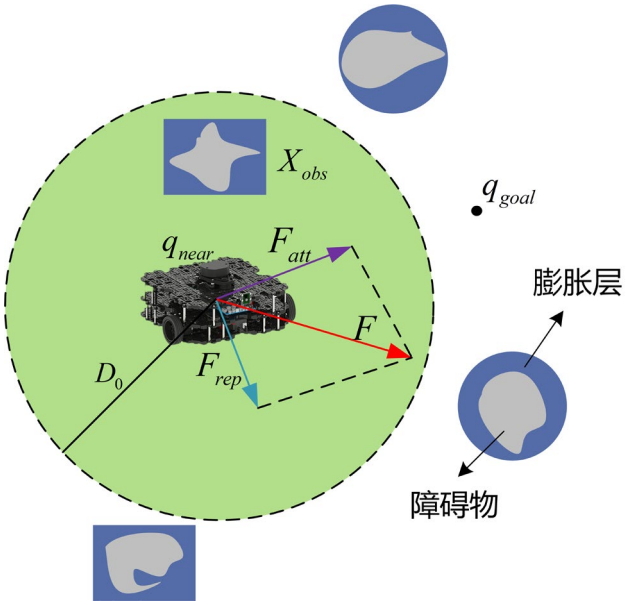


图 3-4 传统人工势场扩展图

但人工势场在复杂环境下存在引力和斥力相互抵消的情况，导致算法陷入局部最优，无法完成路径的扩展。局部最优如图 3-5 所示，此时引力F_{att}和斥力F_{rep}相互抵消，导致合力$F=0$，随机树无法完成扩展，车辆陷入局部最优。为了解决局部最优问题，加快算法的收敛速度，本节设计了一种自适应节点扩展模型。模型使用自适应人工势场对随机树进行扩展，扩展方向始终导向终点。在算法节点扩展的初期使用自适应人工势场对节点进行扩展，如果发现扩展的节点为同一节点即节点扩展陷入局部最优，则使用向量扩展跳出局部最优。

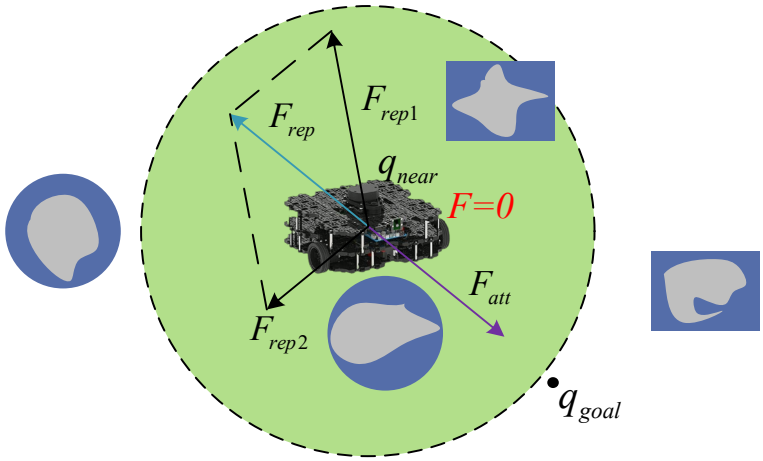


图 3-5 局部最优图

针对复杂环境下，人工势场存在局部最优问题和计算资源大的问题，设计一个选择最优采样方向的函数。从精确采样点 q_{sample} 和随机采样点 q_{rnd} 中选择一个最优采样点方向，在解决局部最优问题的同时加快势场梯度的下降速度。并且自适应的调节斥力场强度，减少算法的计算资源。自适应人工势场原理描述为：

$$U_{\text{att1}} = \frac{1}{2}\gamma D_g(q_{near})^2 \tag{3-11}$$

$$U_{\text{att2}} = \frac{1}{2}\gamma D(q_{sample}, q_{near})^2 \tag{3-12}$$

$$U_{att} = U_{att1} + U_{att2} \tag{3-13}$$

$$F_{att} = -\nabla U_{att} = -\gamma(D_{\text{g}}(q_{near}) + D(q_{sample}, q_{near})) \tag{3-14}$$

$$\kappa = e^{D_0 - D(X_{obs}, q_{near})} \tag{3-15}$$

$$U_{rep} = \begin{cases} \frac{1}{2}\eta \cdot \kappa \cdot (\frac{1}{D(X_{obs}, q_{near})} - \frac{1}{D_0})^2 & , D(X_{obs}, q_{near}) \le D_0 \\ 0 & , D(X_{obs}, q_{near}) > D_0 \end{cases} \tag{3-16}$$

$$Frep = -\nabla U_{rep} = \begin{cases} \eta \cdot \kappa (\frac{1}{D(X_{obs}, q_{near})} - \frac{1}{D_0}) \frac{\nabla D(X_{obs}, q_{near})}{D(X_{obs}, q_{near})^2} & , D(X_{obs}, q_{near}) \le D_0 \\ 0 & , D(X_{obs}, q_{near}) > D_0 \end{cases} \tag{3-17}$$

$$F = F_{att} + \sum_i F_{rep} \tag{3-18}$$

$$q_{new} = q_{near} + step * F \tag{3-19}$$

其中 $D_g(q_{\text{near}})$ 为点 q_{near} 到 q_{goal} 的欧式距离，γ 为引力增益因子，U_{att1} 为终点对点 q_{near} 的引力场，U_{att2} 为精确采样点 q_{sample} 和随机采样点 q_{rand} 中最优方向点对 q_{near} 的引力场，F_{att} 为引力。κ 为自适应因子，$D(X_{obs}, q_{\text{near}})$ 为点 q_{near} 到障碍物 X_{obs} 的欧氏距离，U_{rep} 是障碍物 X_{obs} 对 q_{near} 的斥力场。F_{rep} 为斥力，η 为斥力增益因子，D_0 为斥力的作用范围，$step$ 为扩展步长。F 为合力，扩展新节点的过程如式(3-19)所示。

自适应人工势场如图 3-6 所示，其中紫色实线为点 q_{near} 受到的引力 F_{att}，蓝色实线为点 q_{near} 受到的斥力 F_{rep}，而红色实线是点 q_{near} 受到的合力 F。自适应人工势场算法通过选择最优引力方向加快算法收敛速度，同时使用自适应斥力减少算法的计算复杂度。

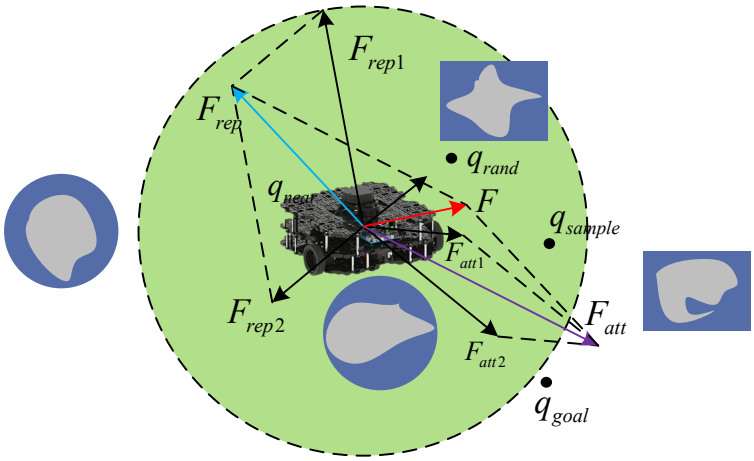


图 3-6 自适应人工势场图

3.2.3 基于贪婪算法剪枝和 B 样条平滑算法的路径优化

虽然 PSRRT*算法生成的全局路径相较于 RRT*算法拐点更少，但是还是存在很多冗余的拐点并且路径的平滑性有待提高。因此，为进一步优化全局路径，需要采用基于贪婪策略的路径剪枝方法，以减少冗余路径点并提升路径质量。在路径优化过程中，首先对路径起点与终点之间的连线进行碰撞检测，以评估其可行性。若该连线未发生碰撞，则可安全地移除路径起点与终点之间的所有中间路径点，从而简化路径结构，提高规划效率。若检测到碰撞，则应继续执行碰撞检测，依次比较路径起点与倒数第二个路径点的连线是否可行，直至找到起点至某一中间路径点间无碰撞的最远连线。在此基础上，对新的起点（即该中间路径点）重复相同的碰撞检测过程，直至所有路径点均完成检测与优化。该方法通过逐步剪枝，有效减少路径冗余，提升路径平滑度与可行性，为后续轨迹优化与车辆运动控制提供更优的路径基础。基于贪婪策略的路径剪枝方法如图 3-7 所示，由 PSRRT*算法规划出的初始路径中包含路径点$(p_1,p_2,p_3,p_4,p_5,p_6,p_7,p_8,p_9)$，经过贪婪策略剪枝优化后的全局路径仅包含路径点$(p_1,p_7,p_9)$，减少了路径代价和拐点数量。

尽管剪枝后的全局路径具有较少的拐点和较低的路径代价，但其优化过程中未充分考虑路径的微分连续性，这可能对车辆运动的平稳性与稳定性产生不利影响，导致轨迹执行过程中出现不必要的速度波动或转向不连续的问题。为改善上述缺陷，本节采用 B 样条曲线对全局路径进行优化与平滑处理，以提高路径的平滑度和可行性，从而增强智能车的运动流畅性与控制稳定性。

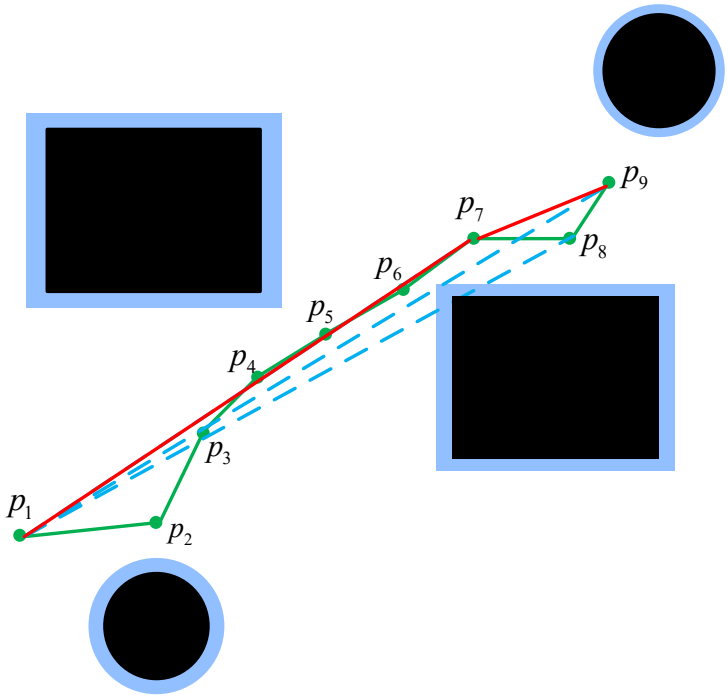


图 3-7 基于贪婪算法剪枝图

B 样条曲线是路径规划领域常用的轨迹优化方法，它通过逼近一组控制点来产生曲线。设全局路径包含 $n+1$ 个控制点 $P_i(i=0,1,\ldots,n)$ 和一个节点向量 $U=\{u_0,u_1,\ldots,u_m\}$，必须满足 $m=n+k+1$，依次连接这些控制点可以构成一条平滑曲线，$k+1$ 阶 B 样条曲线的表达式为：

$$p(u)=\sum_{i=0}^{n}P_iN_{i,k}(u) \tag{3-20}$$

式(3-20)中 $N_{i,k}(u)$ 为 k 次 B 样条基函数，由 Cox-de Boor 递归公式构成，表达式为：

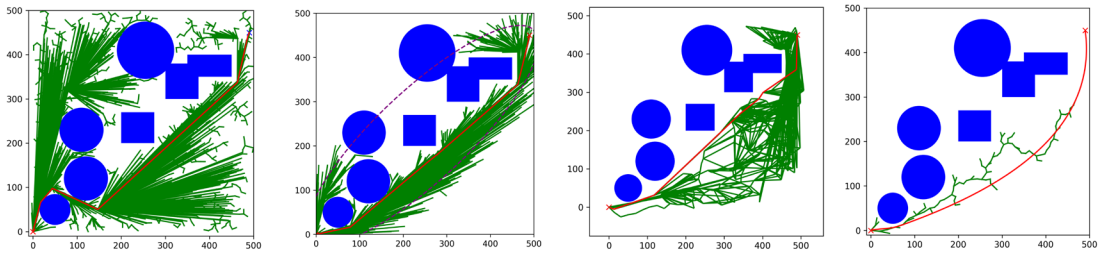
$$\begin{cases} N_{i,0}(u)=\begin{cases}1 & ,u_i\le u\le u_{i+1}\\ 0 & ,\text{其他}\end{cases} \\ N_{i,k}(u)=\dfrac{(u-u_i)N_{i,k-1}(u)}{u_{i+k}-u_i}+\dfrac{(u_{i+k+1}-u)N_{i+1,k-1}(u)}{u_{i+k+1}-u_{i+1}},u_k\le u\le u_{n+1} \end{cases} \tag{3-21}$$

B 样条的高阶连续性可确保路径曲率的平滑变化，有效避免运动突变，而其凸包特性使路径始终约束于控制点定义的安全区域内，从而提升路径规划的安全性与可靠性。

3.3 仿真实验与数据分析

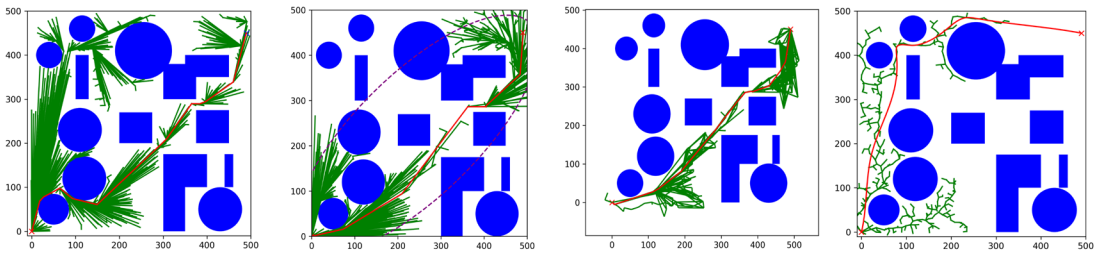
为确保实验结果的可靠性，本研究设计了多维度对比实验：选取 500×500 cm 与 1000×1000 cm 两种地图尺度，并在每种尺度下设置 20%、35%、40%三个障碍物密度等级。在每个实验场景中，分别采用 RRT*算法、Informed RRT*算法[67]、PFRRT*算法[68]以及本节提出的 PSRRT*算法进行 100 次独立实验。基于实验数据，从路径生成代价、算法运行时间和生成的节点数三个性能指标对算法进行系统性评估与分析。图 3-8 至图 3-13 为对应环境下的路径规划结果对比图。

路径规划结果对比图中的蓝色部分为障碍物，白色区域为可行区域，绿色曲线为算法生成的随机树，红色曲线为规划的最终路径。通过对比不同环境下的规划结果可以发现，相较于 RRT*、Informed RRT*和 PFRRT 算法，本节提出的 PSRRT*算法在平滑性和节点效率两个关键指标上均表现出显著优势。



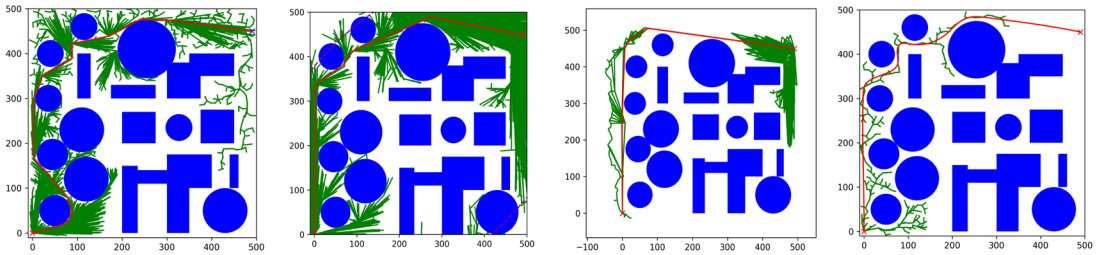
(a) RRT* (b) Informed RRT* (c) PFRRT* (d) PSRRT*

图 3-8 500×500，障碍物占比 20%数据路径规划图



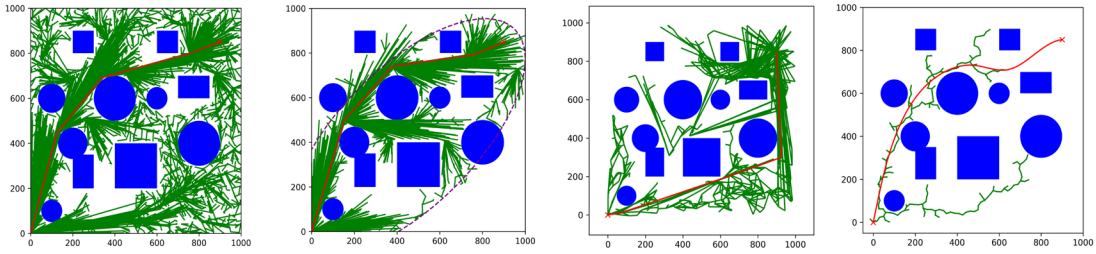
(a) RRT* (b) Informed RRT* (c) PFRRT* (d) PSRRT*

图 3-9 500×500，障碍物占比 35%路径规划图



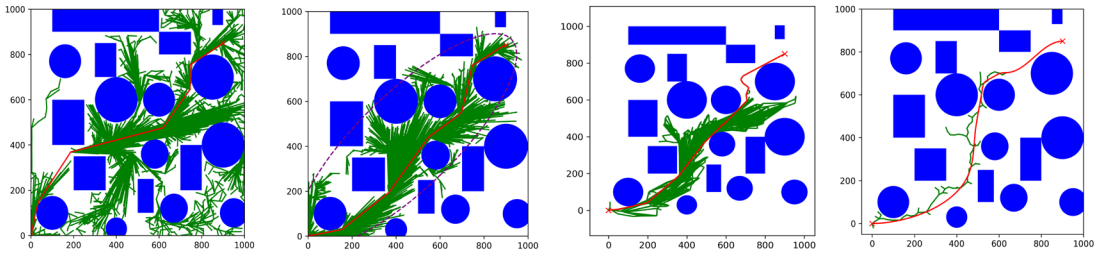
(a) RRT* (b) Informed RRT* (c) PFRRT* (d) PSRRT*

图 3-10 500×500，障碍物占比 40%路径规划图



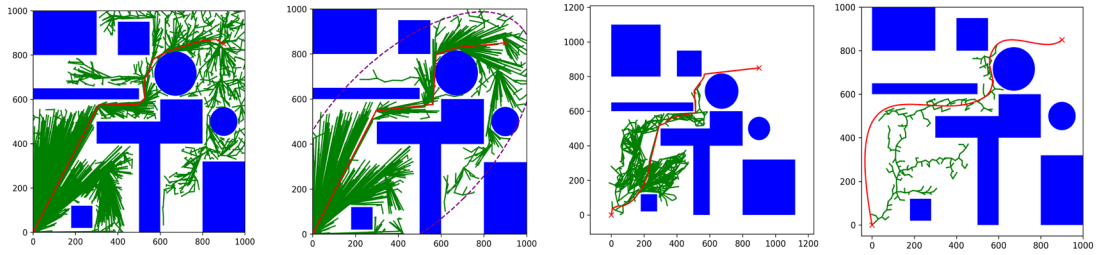
(a) RRT*　(b) Informed RRT*　(c) PFRRT*　(d) PSRRT*

图 3-11 1000×1000，障碍物占比 20%路径规划图



(a) RRT*　(b) Informed RRT*　(c) PFRRT*　(d) PSRRT*

图 3-12 1000×1000，障碍物占比 35%路径规划图



(a) RRT*　(b) Informed RRT*　(c) PFRRT*　(d) PSRRT*

图 3-13 1000×1000，障碍物占比 40%路径规划图

3.3.1 路径代价分析

图 3-14 和表 3-1 分别展示了不同环境条件下 RRT*、Informed RRT*、PFRRT 及 PSRRT*算法的路径代价箱型图对比结果和实验数据。在障碍物密度为 20%的测试环境中，500×500 cm 和 1000×1000 cm 地图尺寸下，RRT*算法的路径代价分别为 820.60、1463.86，Informed RRT*算法的路径代价分别为 813.33、1490.56，PFRRT*算法的路径代价分别为 823.84、1470.26，提出的 PSRRT*算法的路径代价分别为 814.87、1451.45。实验结果表明，PSRRT*算法相较于 RRT*算法路径代价分别减少 0.70%、0.71%，相较于 Informed RRT*算法路径代价分别减少 -0.19%、2.69%，相较于 PFRRT*算法路径代价分别减少 1.09%、1.35%。

随着障碍物密度从 20%提升至 40%，各个算法的路径代价各算法的路径代

价均呈现非线性增长趋势，但 PSRRT*算法的路径代价始终保持较低的水平，这一优势在图 3-14 的箱型图分布差异中得到验证。此外，通过分析正态分布曲线可以发现，PSRRT*算法具有最大的峰度系数和最窄的分布尾部，这表明该算法在路径规划中具有优异的稳定性和广泛的场景适应性。

表 3-1 路径代价仿真数据对比结果

地图尺寸/cm	障碍物密度	路径代价/cm			
		RRT*	Informed RRT*	PFRRT*	PSRRT*
500×500	20%	820.5963	813.3261	823.8361	814.8673
	35%	871.6885	878.5655	889.3611	865.4433
	40%	918.9084	910.4335	959.1127	914.5516
1000×1000	20%	1463.8564	1490.5636	1470.2602	1451.4535
	35%	1492.5246	1368.5365	1461.1566	1398.9358
	40%	1599.7865	1445.4342	1506.7996	1432.7064

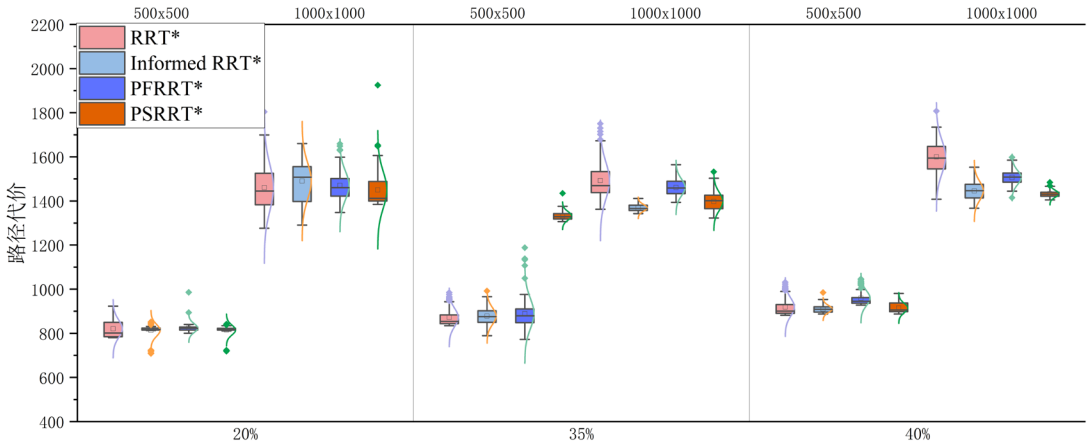


图 3-14 路径代价对比图

3.3.2 运行时间分析

图 3-15 和表 3-2 分别展示了不同环境条件下 RRT*、Informed RRT*、PFRRT*及 PSRRT*算法的运行时间箱型图对比结果和实验数据。在障碍物密度为 20%的场景中，500×500 cm 和 1000×1000 cm 地图尺度下，RRT*算法的运行时间分别为 3.16、4.21，Informed RRT*算法的运行时间分别为 1.77、1.36，PFRRT*算法的运行时间分别为 0.76、1.13，PSRRT*算法的运行时间分别为 0.44、

0.69。实验结果表明，PSRRT*算法相较于 RRT*算法的运行时间分别减少 86.133%、83.63%，相较于 Informed RRT*算法的运行时间分别减少 74.52%、49.21%，相较于 PFRRT*算法的运行时间分别减少 44.08%、38.32%。

随着障碍物密度从 20%逐步提升至 40%，算法规划路径所需的运行时间也在增加， 但 PSRRT*算法的运行时间显著低于其他算法。并且从图 3-15 可以看出，障碍物密度变化不会对 PSRRT*算法的运行时间产生较大影响。这一特性充分证明了 PSRRT*算法具有优异的实时性和鲁棒性，能够满足动态环境下的路径规划需求。

表 3-2 运行时间仿真数据对比结果

地图尺寸/cm	障碍物密度	运行时间/s			
		RRT*	Informed RRT*	PFRRT*	PSRRT*
500×500	20%	3.1563	1.7721	0.7640	0.4382
	35%	6.2950	2.3892	1.5939	0.7448
	40%	11.7355	8.4496	2.6892	1.0968
1000×1000	20%	4.2129	1.3643	1.1265	0.6908
	35%	7.0929	4.3914	2.5461	1.0900
	40%	9.6207	7.0875	3.4605	1.5994

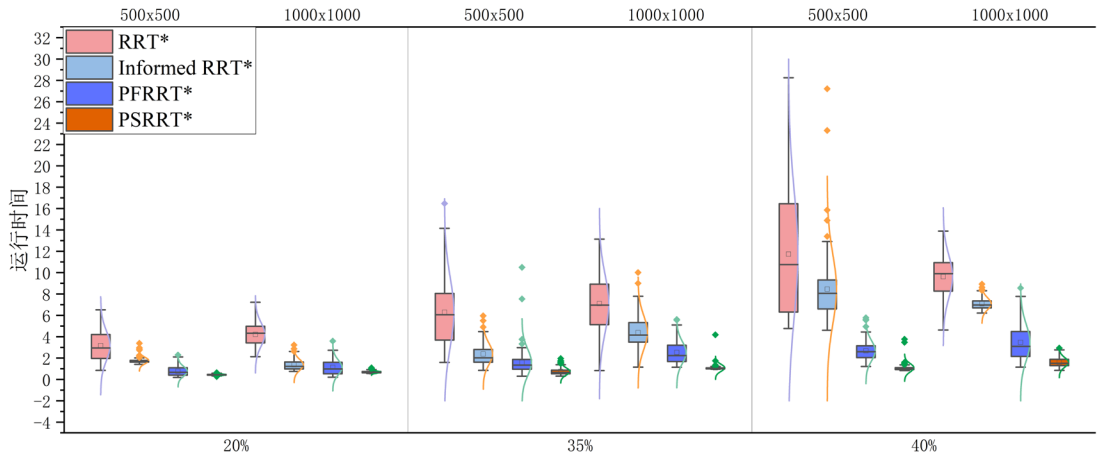


图 3-15 运行时间对比图

3.3.3 节点数分析

图 3-16 和表 3-3 分别展示了不同环境条件下 RRT*、Informed RRT*、

PFRRT*及 PSRRT*算法的运行时间箱型图对比结果和实验数据。在障碍物密度为 20%的场景中，500×500 cm 和 1000×1000 cm 地图尺度下，RRT*算法生成的节点数分别为 1108、910，Informed RRT*算法生成的节点数分别为 1300、944，PFRRT*算法生成的节点数分别为 497、435，PSRRT*算法生成的节点数分别为 236、179。实验结果表明，PSRRT*算法相较于 RRT*算法生成的节点数分别减少 78.70%、80.33%，相较于 Informed RRT*算法生成的节点数分别减少 81.85%、81.04%，相较于 PFRRT*算法生成的节点数分别减少 52.52%、58.85%。从图 3-16 的数据箱型图可以看出，随着障碍物密度的增加，算法生成的节点数开始增加，算法的计算复杂度也逐渐增大，但提出的 PSRRT*算法始终保持着最少的节点数，计算复杂度远低于其他三种算法。

表 3-3 节点数仿真数据对比结果

地图尺寸/cm	障碍物密度	节点数/n			
		RRT*	Informed RRT*	PFRRT*	PSRRT*
500×500	20%	1108	1300	497	236
	35%	1030	1076	429	339
	40%	1074	1267	564	280
1000×1000	20%	910	944	435	179
	35%	1188	1103	645	225
	40%	1565	1431	626	289

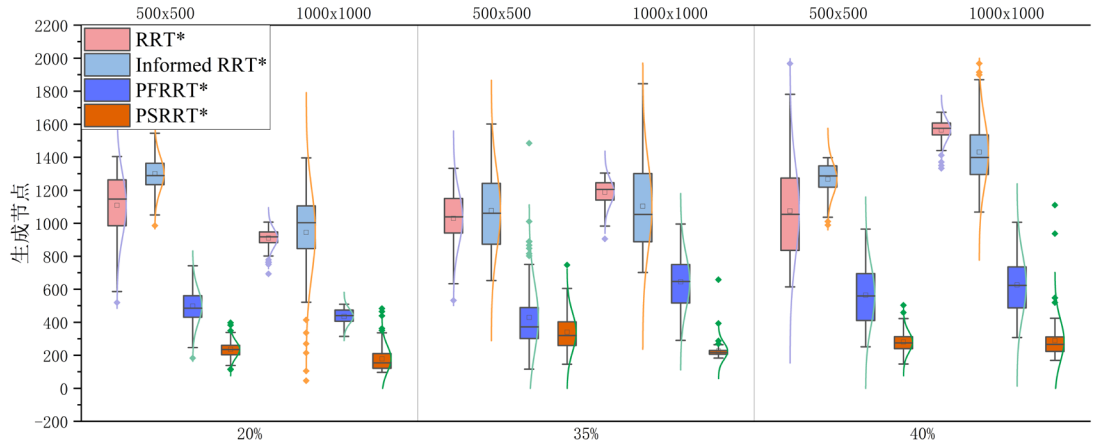


图 3-16 节点数对比图

3.4 本章小结

针对传统的全局路径规划算法在复杂环境下收敛速度慢、难以找到最优路径等问题，本节提出一种 PSRRT*算法。首先使用设计的精确采样模型获取具有目标导向性的采样点，减少无效采样的数量，然后使用自适应节点扩展模型扩展随机树，降低计算资源的消耗，最后对生成的全局路径进行剪枝和平滑处理，减少路径代价。仿真实验表明，本章提出的 PSRRT*算法具有更少的路径代价、更低的运行时间和更少的节点数，综合性能优异。

第4章 基于 BDWAHT 算法的局部路径规划算法研究

虽然 PSRRT*算法在静态环境中能够有效的解决路径规划问题，但是在动态环境下由于无法实时地调整路径，难以保证路径的安全性。为此，本章基于 DWA 算法提出了一种改进的 BDWAHT 算法，并将其与 PSRRT*算法相结合，以增强复杂环境下的路径规划能力。BDWAHT 算法首先通过速度组合预测未来的运动轨迹，然后使用设计的双动态窗口模型优化评估信息，最后使用改进的评估函数选取最佳的预测轨迹作为局部路径。此外，本章基于热传递方程设计了自适应窗口调节机制，以增强算法的鲁棒性。同时，构建了包含未知障碍物和动态障碍物的仿真测试环境，以验证所提算法的性能。

4.1 DWA 算法原理

DWA 算法是一种基于速度空间采样的局部路径规划算法，通过使用采样数据、车辆运动学模型以及其他软约束模拟车辆未来一段时间的最优轨迹，以完成动态环境下的规划任务。DWA 算法在规划轨迹时使用了车辆运动模型，因此规划的路径满足车辆运动学约束，增强了算法的实用性[69]。目前，在智能车的自主避障、导航、探索以及智能仓储中 DWA 算法已得到广泛的应用。该算法主要由速度采样、轨迹预测与筛选两个部分构成，本节将简要介绍这两个部分。

4.1.1 速度采样

DWA 算法通过对车辆设置速度约束获得速度采样范围，随后在该范围内对角速度和线速度进行采样，得到多组符合车辆约束的速度控制条件。车辆的速度约束包括车辆自身物理约束和安全性约束。

车辆自身物理速度 v_l 约束为：

$$v_l = \{(v,\omega)|v \in [v_{\min}, v_{\max}], \omega \in [\omega_{\min}, \omega_{\max}]\} \tag{4-1}$$

其中 v 和 ω 分别为车辆的线速和角速度，$v_{\max}$ 和 $\omega_{\max}$ 分别为车辆的最大线速度和最大角速度，$v_{\min}$ 和 $\omega_{\min}$ 分别为车辆的最小线速度和最小角速度。

安全约束包括加速度约束和障碍物约束。加速度影响下的速度 v_a 约束为：

$$v_a = \{(v,\omega)|v \in [v_s - a_v \cdot \Delta t, v_s + a_v \cdot \Delta t], \omega \in [\omega_s - a_\omega \cdot \Delta t, \omega_s + a_\omega \cdot \Delta t]\} \tag{4-2}$$

式(4-2)中 v_s 和 ω_s 分别为车辆当前状态的线速度和角速度， a_v 和 a_ω 分别为线加速度和角加速度， Δt 为采样时间。

障碍物影响下的速度 v_o 约束为：

$$v_o = \{(v,\omega) | v \in [0, \sqrt{a_v \cdot 2d_o(v,\omega)}], \omega \in [0, \sqrt{a_\omega \cdot 2d_o(v,\omega)}]\} \tag{4-3}$$

其中 $d_o(v,\omega)$ 为车辆位置到障碍物距离的最小值。综上对车辆速度采样范围的限制，车辆速度 v_v 采样范围为：

$$v_v = v_l \cap v_a \cap v_o \tag{4-4}$$

车辆速度采样值 (v_s, ω_s) 为：

$$(v_s, \omega_s) = (v_{min} + i \cdot \Delta v, \omega_{min} + j \cdot \Delta\omega) \tag{4-5}$$

式(4-5)中 $i = 0,1,2,\ldots,\frac{v_{\max} - v_{\min}}{\Delta v}$， $j = 0,1,2,\ldots,\frac{\omega_{max} - \omega_{\min}}{\Delta\omega}$， Δv 和 $\Delta\omega$ 分别为线速度采样间隔和角速度采样间隔。

4.1.2 轨迹预测与筛选

车辆的运动学模型如式(2-1)所示，利用时间步长 Δt，通过欧拉法离散化得到车辆轨迹预测式(4-6)，由于此时存在角速度 ω，因此航向角 φ 可由 ω 表达。

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \varphi_{t+1} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \varphi_t \end{bmatrix} + \begin{bmatrix} cos(\varphi_t) \cdot v \\ sin(\varphi_t) \cdot v \\ \omega \end{bmatrix} \cdot \Delta t \tag{4-6}$$

其中 v 、 ω 、 Δt 分别为车辆线速度、车辆角速度、预测时间间隔。

根据式(4-5)速度采样值和式(4-6)预测车辆未来 T 时间内的轨迹。但是由于存在多组预测轨迹，如何快速的从多条预测轨迹中选择最优轨迹，这对于路径规划的性能有着重大的影响。DWA 算法使用轨迹评价函数 $G(v,w)$ 来选择最优轨迹，以提高路径规划的性能。轨迹评价函数包含三个子函数 $h(v,w)$ 航向角评价子函数，$d(v,w)$ 距离评价子函数，$v(v,w)$ 速度评价子函数。轨迹评价函数 $G(v,w)$ 如式(4-10)所示。

$$h(v,\omega) = \pi - \vartheta \tag{4-7}$$

$$d(v,\omega) = \min_{i=1,2,\ldots,n} \left(\sqrt{(x_v - x_i)^2 + (y_v - y_i)^2} - r_i \right) \tag{4-8}$$

$$v(v,\omega) = |v| \tag{4-9}$$

$$G=\sigma(\alpha\cdot h(v,\omega)+\beta\cdot d(v,\omega)+\gamma\cdot v(v,\omega)) \tag{4-10}$$

其中 ϑ 为车辆当前位置与目标点间的夹角，(x_v,y_v) 为车辆的当前坐标，(x_i,y_i,r_i) 为第 i 个障碍物配置。$\sigma(\cdot)$ 为归一化处理，因三个子函数的物理意义不同需要进行归一化后才能进行评价，α、β、γ 为对应的权重。

4.2 BDWAHT 算法

针对复杂动态环境下 DWA 算法规划时间长、计算复杂度大等问题，本节提出一种 BDWAHT 算法，以增强复杂动态环境下局部路径规划算法的实时性。首先，针对传统 DWA 算法规划时间长的问题，提出一种双动态窗口模型。在双动态窗口模型中，设计了障碍物窗口，并将其与速度窗口进行融合，以加快算法的规划速度。其次，为了增强算法的适用性并减少计算复杂度，基于热传递方程设计了窗口自适应调节模型。在该模型中，将障碍物模拟为不断释放热量的热源，车辆根据周围环境的热量变化来自适应的调节障碍物窗口的大小，减少计算时间。最后，使用改进的轨迹评价函数选取最优轨迹，完成局部路径规划任务。BDWAHT 算法框图如图 4-1 所示。

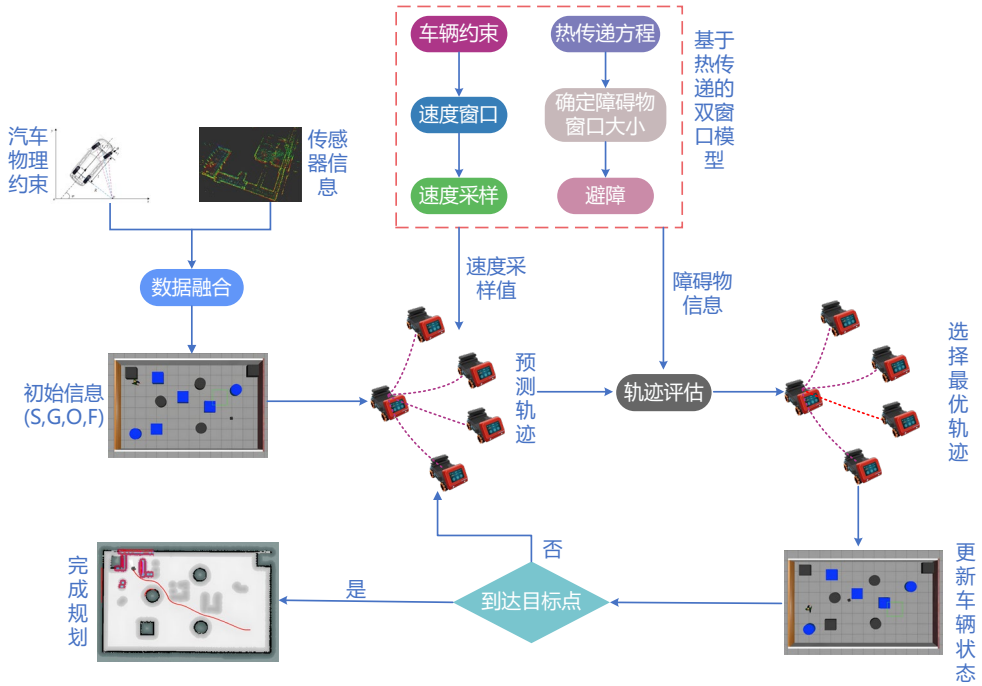


图 4-1 BDWAHT 算法

4.2.1 双动态窗口模型

双动态窗口模型包括速度动态窗口和障碍物动态窗口。速度动态窗口是基于车辆运动学特性和安全约束条件所确定的速度可行域，通过限制速度搜索空间来降低路径规划的计算复杂度，同时确保行驶过程的安全性。速度动态窗口如图4-2所示。

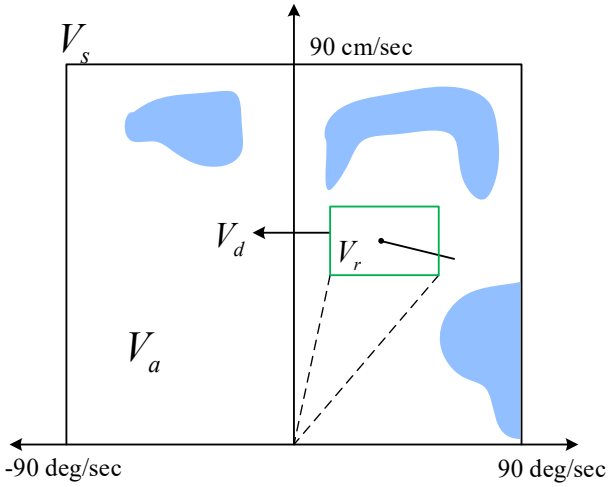


图 4-2 速度动态窗口示意图

其中浅蓝色部分为障碍物，绿色矩形为速度动态窗口，V_s 为车辆物理约束下的速度范围，V_a 为加速度约束下的速度范围，V_d 为障碍物约束下的速度范围，V_r 为速度动态窗口的范围。

根据车辆自身的物理驱动电机的约束，速度 V_s 约束为：

$$V_s = \{(v,\omega) | V \in [v_{\min}, v_{\max}], \omega \in [\omega_{\min}, \omega_{\max}]\} \tag{4-11}$$

其中 v 和 ω 分别为车辆的线速和角速度，$v_{\max}$ 和 $\omega_{\max}$ 分别为车辆的最大线速度和最大角速度，$v_{\min}$ 和 $\omega_{\min}$ 分别为车辆的最小线速度和最小角速度。

加速度影响下的速度 V_a 约束为：

$$V_a = \{(v,\omega) | v \in [v_s - a_v \cdot \Delta t, v_s + a_v \cdot \Delta t], \omega \in [\omega_s - a_\omega \cdot \Delta t, \omega_s + a_\omega \cdot \Delta t]\} \tag{4-12}$$

其中 v_s 和 ω_s 分别为车辆当前状态的线速度和角速度，a_v 和 a_ω 分别为线加速度和角加速度，Δt 为采样时间。

障碍物约束下的速度 V_d 约束为：

$$V_d = \{(v,\omega) | v \in [0, \sqrt{a_v \cdot 2d_o(v,\omega)}], \omega \in [0, \sqrt{a_\omega \cdot 2d_o(v,\omega)}]\} \tag{4-13}$$

其中 $d_o(v,\omega)$ 为车辆位置到障碍物距离的最小值。综上对车辆速度采样范围的限制，速度动态窗口 V_r 采样范围为：

$$V_r = V_s \cap V_a \cap V_d \tag{4-14}$$

传统的动态窗口只有一个动态窗口即速度动态窗口，在进行轨迹评估时会对所有的障碍物进行安全性评估。但是有的障碍物实际上不会对车辆的安全性造成影响，此时对这些障碍物进行安全性评估增加了算法的计算量，导致规划时间增加。部分障碍物影响如图 4-3 所示，红色圆点为车辆的当前位置，黑色圆点为路径规划终点，其中的三个紫色障碍物对最优路径的安全性没有影响，此时若评估这些障碍物对最优路径的影响没有意义。同时这些障碍物距离车辆的位置较远，如果在评估轨迹时将其参与计算会增加计算复杂度，延长规划时间。

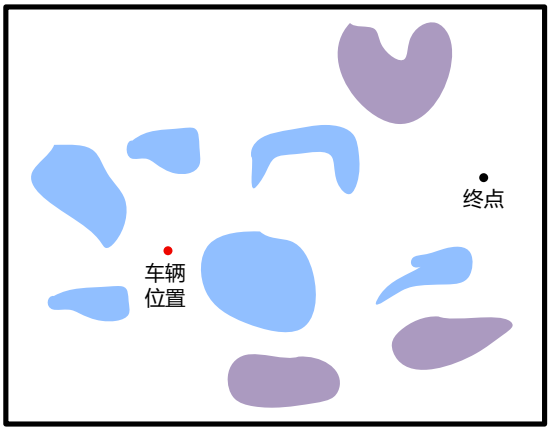


图 4-3 部分障碍物影响示意图

为了解决上面的问题，本节除了速度动态窗口外还设计了一个障碍物动态窗口，用于减少计算量，加快规划速度。障碍物动态窗口示意图如图 4-4 所示，其中紫色矩形为障碍物动态窗口，车辆位于动态窗口的中心，只有在障碍物动态窗口内的障碍物才参与轨迹评估的计算。

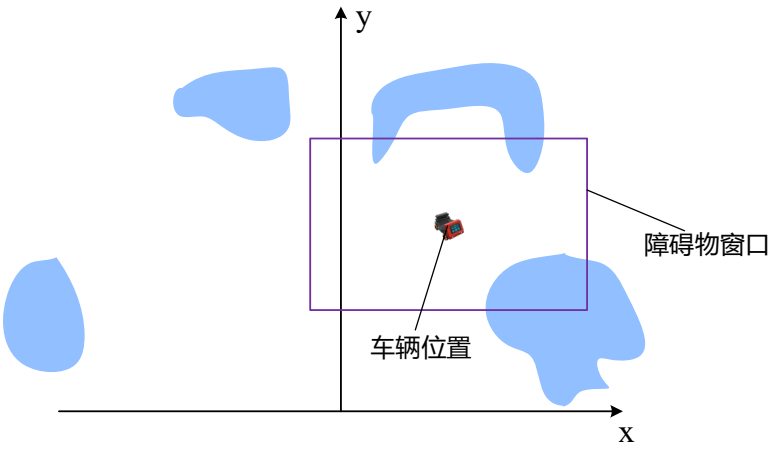


图 4-4 障碍物动态窗口示意图

基于速度动态窗口和障碍物动态窗口的特点，本节构建了如图 4-5 所示的双动态窗口模型。其中绿色矩形为速度动态窗口，紫色矩形为障碍物动态窗口，红色虚线为在双窗口约束下算法预测的车辆未来轨迹。模型首先在速度空间内对速度动态窗口进行采样，获取速度采样值并生成车辆预测轨迹。然后，通过分析状态空间中障碍物动态窗口内的障碍物分布及其对车辆预测轨迹的影响，算法能够提前舍弃不安全的预测结果。这种双重约束机制不仅保证了预测轨迹的安全性，同时也显著降低了算法的计算复杂度。

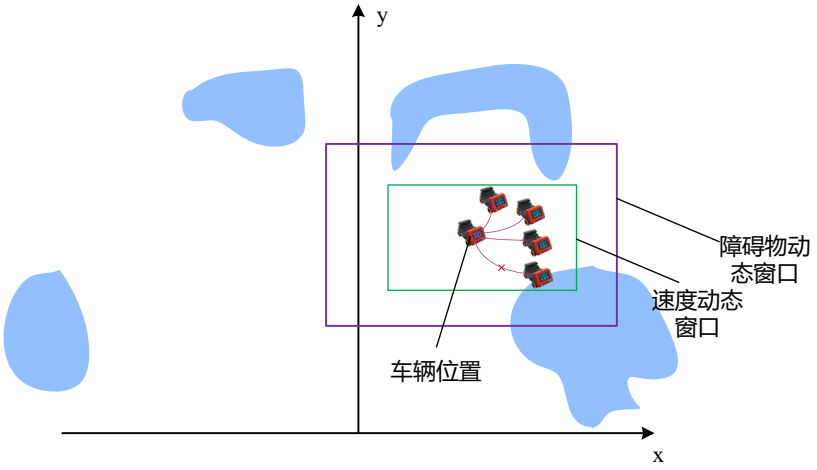


图 4-5 双动态窗口模型

双动态窗口模型可以有效地降低算法的计算复杂度，加快算法的规划速度。同时由于该模型将速度动态窗口和障碍物动态窗口进行了融合，所以并不会降低原有的规划精度。

4.2.2 基于热传递的自适应窗口调节

虽然双动态窗口模型能有效地降低计算复杂度，加快收敛速度，但是其固定宽度的窗口尺寸在应对复杂环境时仍存在局限性。传统固定窗口尺寸的算法在面对复杂环境时，往往由于窗口尺寸的约束而导致两种情况：窗口过大造成计算资源浪费；窗口过小导致搜索范围不足。针对这一问题，本节基于热传递方程设计了自适应窗口调节机制。该机制的核心思想是将障碍物类比为热源，通过建立热传递方程来量化窗口区域内的热量强度变化，进而实现窗口尺寸的自适应调节。这种基于物理场理论的调节方法能更好的适应复杂动态环境的变化需求。

算法首先对环境中的所有障碍物进行几何膨胀处理，并将其最小外接圆定义

为等效热源。设热量均匀扩散，热源的圆心为$o(x_o,y_o)$，热源半径为r，接收点（车辆）A位置为(x_c,y_c)。

设热源上任意一点(x_i,y_i)为：

$$(x_i,y_i)=(x_o+\rho cos\theta,y_o+\rho sin\theta) \tag{4-15}$$

其中$\rho\in[0,r]$为热源径向距离，$\theta\in[0,2\pi]$为角度。

故接收点（车辆）到该点的距离D为：

$$D=\sqrt{(x_c-(x_o+\rho cos\theta))^2+(y_c-(y_o+\rho sin\theta))^2} \tag{4-16}$$

设热源上的总热量Q均匀分布，单位面积上的热量δ为：

$$\delta=\frac{Q}{\pi r^2} \tag{4-17}$$

单位释放的热量dQ为：

$$dQ=\delta\cdot\rho\cdot d\rho d\theta=\frac{Q\cdot\rho}{\pi r^2}d\rho d\theta \tag{4-18}$$

故热量强度H为：

$$dH=\frac{dQ}{2\pi D}=\frac{Q\cdot\rho}{2\pi^2 r^2 D}d\rho d\theta \tag{4-19}$$

$$H=\frac{Q}{2\pi^2 r^2}\int_0^{2\pi}\int_0^r\frac{\rho}{D}d\rho d\theta \tag{4-20}$$

总热量强度H_s为：

$$H_s=\sum_{i=0}^{n}H_i \tag{4-21}$$

窗口宽度l_w为：

$$l_w=\begin{cases}1.5\cdot l_0 & ,H_s>H_0\\ \dfrac{H_s}{H_0}\cdot l_0+o & ,H_s\le H_0\end{cases} \tag{4-22}$$

其中H_0为初始热量强度，l_0为设定的最小窗口宽度，o为最小窗口尺寸。

基于热传递的动态窗口如图 4-6 所示，紫色矩形为障碍物动态窗口，蓝色区域为动态窗口外的障碍物，黑色虚线为窗口内障碍物，窗口内不同颜色代表不同的热量强度。窗口内的障碍物（等效热源）不断向外释放热量，算法通过计算热流强度自适应调整窗口大小。

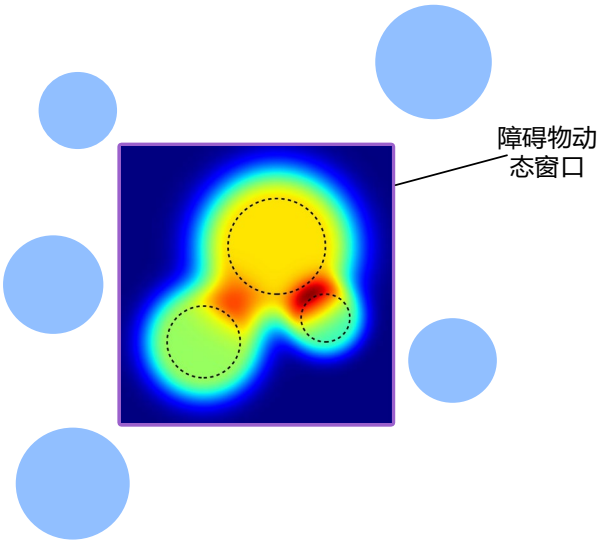


图 4-6 基于热传递的动态窗口

4.2.3 轨迹评价函数

通过前面对 PSRRT*算法的分析可以发现，PSRRT*算法在静态环境中表现出优异全局规划能力，能够规划出路径代价低、安全性高、收敛速度快的路径。但是该算法在动态环境中的实时避障能力明显不足，难以满足智能车在动态环境中的规划需求。另一方面，虽然 DWA 算法在动态环境下有良好的实时避障能力，但是在复杂环境中容易陷入局部最优。并且在全局路径与局部路径融合时，DWA 算法的传统轨迹评价函数会导致路径波动和收敛速度慢等问题。针对上述算法在动态环境中的局限性，本节将 PSRRT*算法与 BDWAHT 算法有机结合，构建了一种新的轨迹评价函数 $G_{new}(v,\omega)$，如式(4-23)所示。

$$G_{new}=\sigma(\alpha\cdot h(v,\omega)+\beta\cdot d(v,\omega)+\gamma\cdot v(v,\omega)+\eta\cdot g(v,\omega)+\mu\cdot p(v,w)) \quad (4\text{-}23)$$

式(4-23)中 $h(v,\omega)$、$d(v,\omega)$、$v(v,\omega)$ 为传统轨迹评估函数中的评估子函数，$g(v,\omega)$ 为目标方向子函数，$p(v,\omega)$ 为路径平滑评估子函数，α、β、γ、η、μ 为对应子函数的权重。

在融合全局路径时，传统的轨迹评估函数仅使用航向评估函数 $h(v,\omega)$ 来衡量车辆与当前全局路径点的对齐情况，这种单点评估机制忽略了下一个全局路径点的位置信息，这就导致了路径波动。为了解决这个问题，本节通过建立当前全局路径点、预测轨迹终点和下一全局路径点之间的几何关系设计了目标方向子函数 $g(v,\omega)$，如式(4-24)所示，以实现更为精确的轨迹评估。

虽然平滑后的全局路径具有良好的整体平滑性，但其静态特性难以适应动态

环境的变化，难以保证局部路径的平滑性。为了提高局部路径的平滑度并保障车辆行驶的舒适性，本节设计了路径平滑评估子函数 $p(v,\omega)$。该函数通过最小化局部路径和全局路径间的航向角偏差来实现路径平滑，如式(4-25)所示。

$$g(v,\omega)=-\sqrt{(x_{next}-x_{tarE})^2+(y_{next}-y_{tarE})^2} \tag{4-24}$$

$$p(v,\omega)=\begin{cases}-|\,arctan(\dfrac{y_{next}-y_{tarE}}{x_{next}-x_{tarE}})-\varphi_{tarE}\,|, q_{next}=q_{current}\\ -|\,arctan(\dfrac{y_{next}-y_{current}}{x_{next}-x_{current}})-\varphi_{tarE}\,|, others\end{cases} \tag{4-25}$$

其中 φ_{tarE} 为车辆航向角，$q_{current}$ 和 q_{next} 分别当前的全局路径点和下一个全局路径点。(x_{tarE},y_{tarE}) 为预测轨迹的终点坐标，$(x_{current},y_{current})$ 为 $q_{current}$ 点坐标，(x_{next},y_{next}) 为 q_{next} 点坐标。$arctan(\bullet)$ 为反正切函数。

4.3 融合算法流程

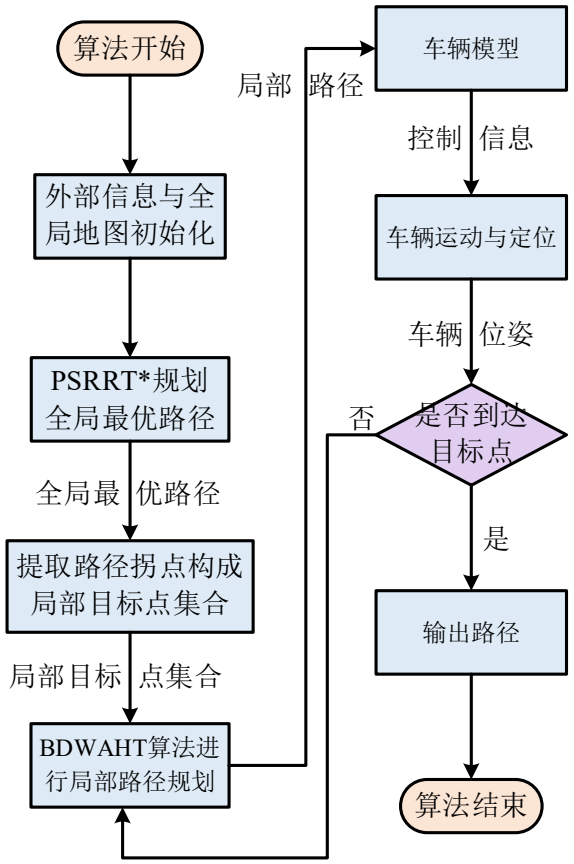


图 4-7 PSRRT*-BDWAHT 融合算法流程图

第三章提出的 PSRRT*算法能够在静态环境下生成全局最优路径，但由于其缺乏实时调整路径的能力，难以应对未知障碍物或动态障碍物。BDWAHT 算法

虽可加快局部路径的收敛速度，然而作为局部路径规划算法，其无法获取全局路径信息，可能导致次优路径的生成。因此，本章融合 PSRRT*与 BDWAHT 算法，以兼顾全局最优性与实时避障能力，使路径规划既具备全局优化特性，又能有效避开未知及动态障碍物。融合算法流程图如图 4-7 所示。

首先，融合算法初始化外部环境信息与全局地图，为 PSRRT*算法的全局路径规划提供基础数据支持。随后，采用 PSRRT*算法生成全局最优路径，并提取路径拐点以构建局部目标点集合。最后，BDWAHT 算法依次对各局部目标点进行路径规划，若检测到车辆位姿已到达目标点，则终止规划并输出最终路径；否则，继续针对下一个局部目标点进行路径优化，直至路径规划完成。

4.4 仿真实验与数据分析

为了验证提出 PSRRT*-BDWAHT 算法的有效性，本文使用 Intel(R) Core(TM) i7-8550U CPU 和 8G RAM 的计算机，在 Visual Studio Code 软件上分别在静态环境和动态环境下对智能车进行避障仿真分析。

4.4.1 静态环境下仿真实验分析

为了验证提出的 PSRRT*-BDWAHT 算法在静态环境中对未知障碍物的避障能力，本节在静态环境中添加未知障碍物。初始完全已知的静态地图如图 4-8 所示。

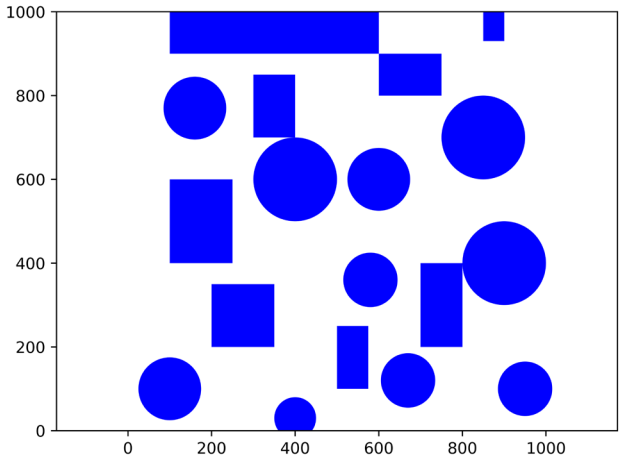


图 4-8 静态地图

使用 PSRRT*算法规划的全局路径如图 4-9 所示。

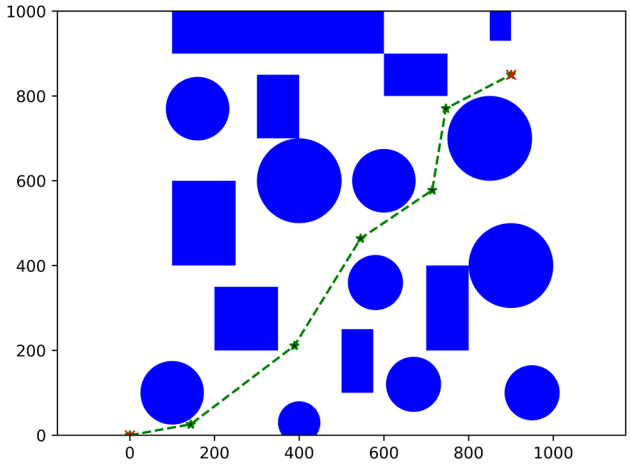


图 4-9 全局路径规划结果

图中(0,0)为路径起点，(900,850)为路径终点，绿色虚线为规划的全局路径，绿色*为局部目标点。在原来的静态地图上(250,110)、(440,230)、(760,560)处增加未知障碍物，分别使用 PSRRT*-DWA 算法和 PSRRT*-BDWAHT 算法进行动态规划，规划对比如图 4-11 所示，实验对比数据如表 4-1 所示，数据对应的雷达图如图 4-10 所示。

表 4-1 动态规划实验数据

环境	PSRRT*-DWA 算法			PSRRT*-BDWAHT 算法		
	运行时间	路径代价	平滑度	运行时间	路径代价	平滑度
静态环境	70.67	1322.61	7.38	47.86	1312.60	6.98
动态环境	102.41	1332.57	9.23	51.96	1322.48	7.40

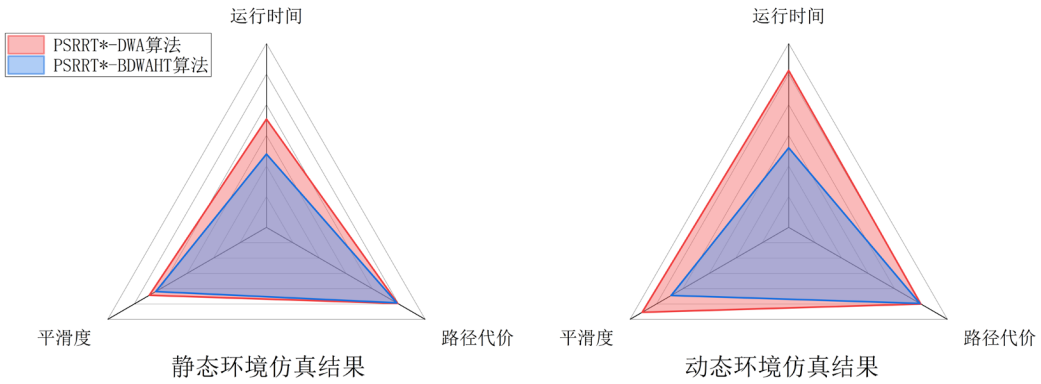
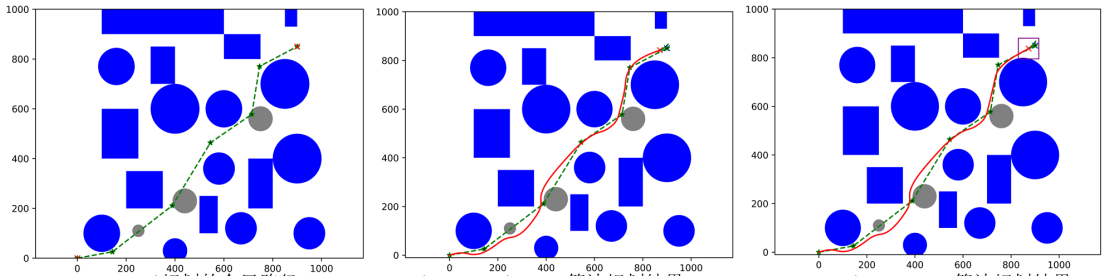


图 4-10 仿真数据雷达图

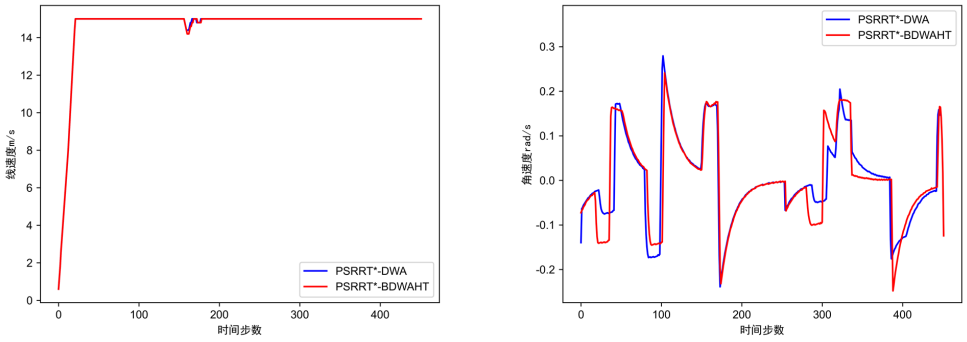
图 4-11 中绿色虚线为 PSRRT*算法规划的全局路径，灰色区域为未知的静态障碍物，蓝色区域为已知的静态障碍物，红色曲线为最终规划的路径。从动态

规划效果图可以看出，PSRRT*-DWA 算法和 PSRRT*-BDWAHT 算法对于存在未知障碍物的静态环境均能完成规划和避障的任务，而 PSRRT*-BDWAHT 算法规划的路径相较于 PSRRT*-DWA 算法规划的路径更平滑一些。同时从表 4-1 的实验数据可以看出，本节的算法相较于 PSRRT*-DWA 算法在运行时间上提升了 32.27%，在路径的平滑度上提升了 5.42%。实验规划效果图和实验数据表明，PSRRT*-BDWAHT 算法具有更好的实时性、路径平滑性，在复杂静态环境下的规划性能优于 PSRRT*-DWA 算法。



(a) PSRRT*规划的全局路径　(b) PSRRT*-DWA算法规划结果　(c) PSRRT*-BDWAHT算法规划结果

图 4-11 静态环境规划效果对比图



(a) 线速度对比　(b) 角速度对比

图 4-12 静态环境下算法速度对比图

图 4-12 (a)为两种算法的线速度对比图，从图中可以看出两种算法在静态环境下均可以满速运行。图 4-12 (b)为两种算法的角速度对比图，从图中可以看出所提的 PSRRT*-BDWAHT 算法的角速度波动小于 PSRRT*-DWA 算法。

4.4.2 动态环境仿真实验分析

本小节在静态环境地图的基础上增加未知的动态障碍物，以验证提出的 PSRRT*-BDWAHT 算法在复杂动态环境下的规划性能。在静态环境地图的基础上设置动态障碍物，障碍物坐标为(470,80)、(730,450)、(730,970)，x 方向运

动速度分别为-1.5m/s、-1.5m/s、1m/s，y 方向运动速度分别为 1.5m/s、0m/s、-1m/s。PSRRT*-DWA 算法和 PSRRT*-BDWAHT 算法避障和规划过程如图 4-13 所示。

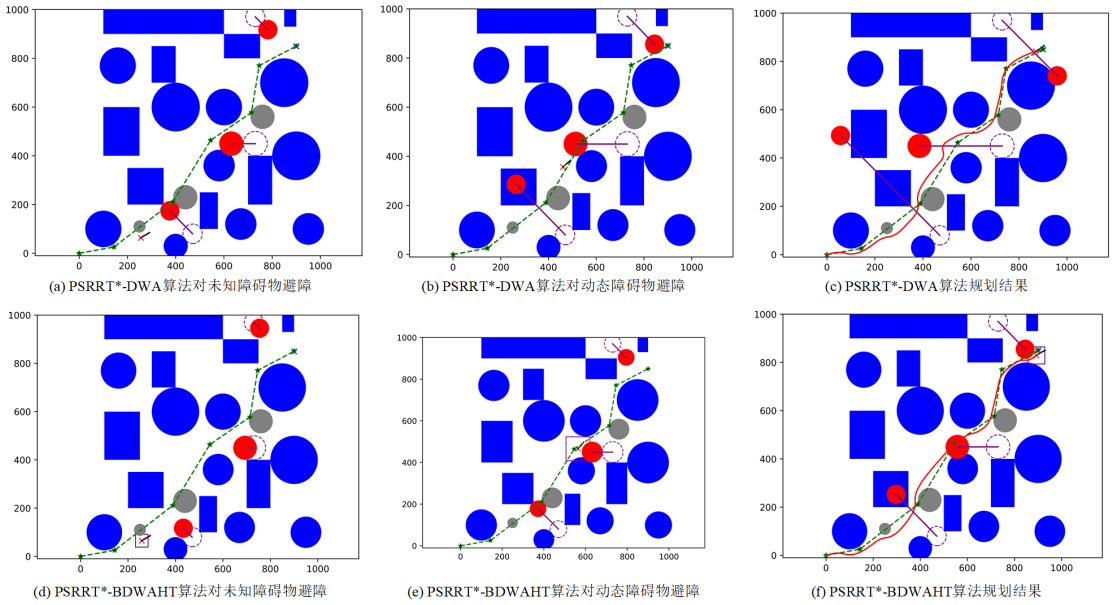
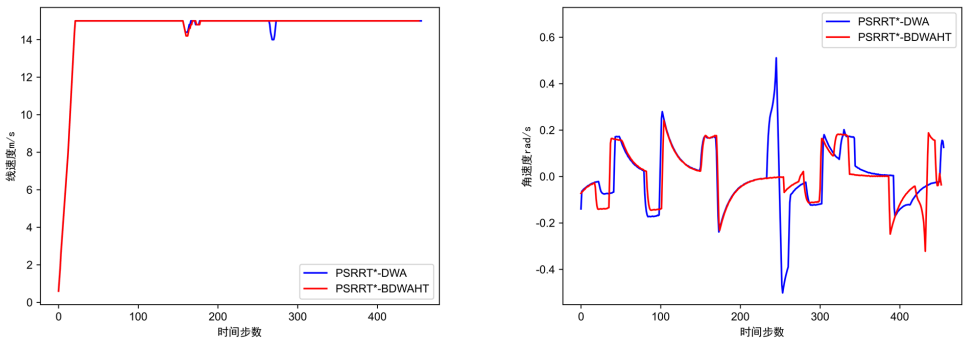


图 4-13 动态环境下避障及规划效果对比图



(a) 线速度对比　　(b) 角速度对比

图 4-14 动态环境下算法速度对比图

图 4-13 中的紫色虚线圆为动态障碍物的起始位置，红色区域为当前时刻的动态障碍物位置，紫色实线为动态障碍物的运动轨迹，红色曲线为规划的路径，紫色矩形为障碍物动态窗口。图 4-13 (a)和图 4-13 (d)为遇到未知的静态障碍物的避障过程，图 4-13 (b)和图 4-13 (e)为遇到动态障碍物的运动过程，图 4-13 (c)和图 4-13 (f)为最终的规划结果。虽然 PSRRT*-DWA 算法也能在复杂动态环境下完成规划与避障任务，但是其规划时间长，路径平滑度低。同时 PSRRT*-DWA 算法的前瞻性较低，如图 4-13 (b)，此时 PSRRT*-DWA 算法无法绕过动态障碍

物，只能原地等待，进一步增加了规划时间。

图 4-14 为动态环境下两种算法的速度对比图，从图中可以看出，PSRRT*-DWA 算法在该环境下有速度波动，而 PSRRT*-BDWAHT 算法没有太大的速度波动，能以更稳定、更快的速度行驶；PSRRT*-DWA 算法的角速度在动态环境下波动较大，而 PSRRT*-BDWAHT 算法有着稳定的角速度。以上的规划结果图和实验数据表明，本节提出的 PSRRT*-BDWAHT 算法在复杂的动态环境下能够更加快速、稳定的完成规划与避障任务。

从表 4-1 的实验数据可以看出，PSRRT*-BDWAHT 算法在复杂的动态环境运行时间仅需 51.96s，相较于 PSRRT*-DWA 算法在运行时间上提升了 49.26%，路径的平滑度为 6.98，相较于 PSRRT*-DWA 算法在路径的平滑度上提升了 19.83%。PSRRT*-BDWAHT 算法在复杂的动态环境运行时间相较于静态环境仅提升了 7.89%，而 PSRRT*-DWA 算法运行时间提升了 30.99%。同样的 PSRRT*-BDWAHT 算法的路径平滑度仅提升了 5.67%，PSRRT*-DWA 算法提升了 20.04%。当环境剧烈变化时，DWA 算法在规划效率和规划效果间无法很好地平衡，算法更偏向于选择规划效率。本章提出的 BDWAHT 算法通过设计的双动态窗口模型、窗口调节机制、新的轨迹函数能够在规划效率和规划效果取得较好的平衡。

4.5 本章小结

为提升 PSRRT*算法在动态环境中的实时性与鲁棒性，本章首先对 DWA 算法的原理进行了简要的介绍，然后针对 DWA 算法在动态环境的问题提出了一种 BDWAHT 算法。在 BDWAHT 算法中首先建立双动态窗口模型减少计算复杂度，然后基于热传递方程自适应的调整动态窗口的大小，增强算法的泛用性，最后使用改进的轨迹评价函数将 PSRRT*算法与 BDWAHT 算法融合。仿真实验与数据分析表明，本章提出的 PSRRT*-BDWAHT 算法不仅增强了局部规划的实时性，而且提升了路径的平滑性，使车辆的行驶更加安全舒适。

第5章 基于 ROS 平台的智能车实验测试

第4章提出的PSRRT*-BDWAHT算法在理想环境测试中表现出优异的性能，为了全面评估算法在实际环境中的性能，本章搭建了基于 ROS 平台的仿真测试与实际环境下的实车实验。本章对 ROS 平台进行了简要的介绍，其次搭建了智能车的实验平台，最后对本文提出的算法进行仿真测试和实车测试，分析其性能，验证算法的有效性。

5.1 ROS 平台

智能车是一种结构复杂、零部件众多且各模块之间紧密耦合的装置。此外，智能车所处的外部测试环境通常是动态多变且充满不确定性的。为更高效地完成智能车的结构设计与实验测试，机器人操作系统（Robot Operating System，ROS）平台应运而生。

ROS 是一款用于智能车、机器人研究与开发的开源操作系统。自 2007 年推出以来，ROS 凭借其强大的功能和灵活的架构，成为智能车领域的重要研究工具。ROS 集通信、服务、文件系统与开源社区于一体，为开发者提供了一个全面的研发框架，显著降低了智能车和机器人系统开发的复杂性。

5.1.1 ROS 的架构

ROS 的架构主要包括三个核心模块：计算图层（Computation Graph Level）、文件系统层（Filesystem Level）和开源社区（Open Source Community）。计算机图层负责分布式节点之间的通信与协作；文件系统层提供了资源管理与工具链支持；开源社区则为开发者提供了丰富的开源资源与技术支持。以下章节将对这三个模块分别进行详细介绍。

（1）计算机图层

在 ROS 系统中，智能车的各项功能被设计为独立的节点。每个节点通过特定的通信机制进行信息交互，共同构建了一张复杂的网状结构，称为计算图（Computation Graph）。该计算图是 ROS 通信架构的核心，负责协调节点之间的数据传输与功能协作，为智能车实现高效信息处理提供了基础。

计算图的主要组成模块包括：

节点（Node）：ROS 系统的基本运行单元，每个节点负责完成特定的功能模块，例如传感器数据采集、路径规划或控制命令生成。

话题（Topic）：节点之间进行数据发布与订阅的通信渠道，支持异步传输，常用于大规模的数据交换，如雷达点云和图像数据交换。

服务（Service）：提供同步的双向通信机制，适合短时间请求和响应场景，如查询传感器状态或执行单次运动命令。

消息（Message）：节点间交换的具体数据格式，通常包含结构化的信息，例如位置、速度或传感器信息。

ROS 计算机图层的整体结构如图 5-1 所示，展示了节点、话题、服务和消息等模块的协作关系，以实现复杂系统的高效运行。

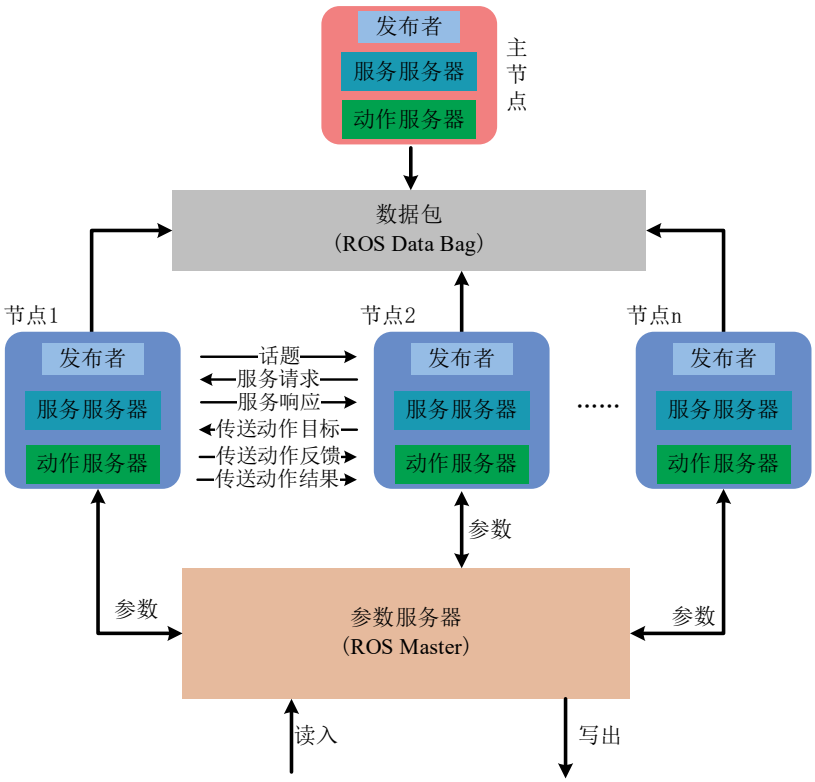


图 5-1 ROS 计算机图层

（2）文件系统层

ROS 的文件系统层承担着资源管理、组织工具链和通信接口定义等关键功能。该层级通过标准化的文件系统结构和命名规范，为开发者提供了统一的资源

存储、检索和管理机制，这种结构化的设计不仅支持模块化的软件开发，同时也为开发者协作提供了必要的框架支持。文件系统层的主要组成模块包括：

功能包：ROS 文件系统层的核心单元，每个包都是一个独立的功能模块，包含实现特定功能的所有资源。

描述文件：每个包的元数据文件，包含依赖关系、许可证等关键信息。

工作空间：工作空间是开发者进行 ROS 项目开发、调试和运行的主要环境。

消息定义和服务定义：ROS 中定义节点间通信格式的重要资源，消息文件用于定义话题通信中数据的结构，服务文件用于定义服务通信请求和响应的格式。

功能包是文件系统层的核心，所有功能的开发、资源的存储都围绕功能包展开。描述文件连接功能包与外部依赖，确保构建和运行时的依赖解析。工作空间将多个功能包整合在一起，为开发者提供统一的开发、测试环境。消息定义和服务定义为通信提供统一规范，节点通过这些规范与其他节点的协同工作。ROS 的文件系统层结构如图 5-2 所示。

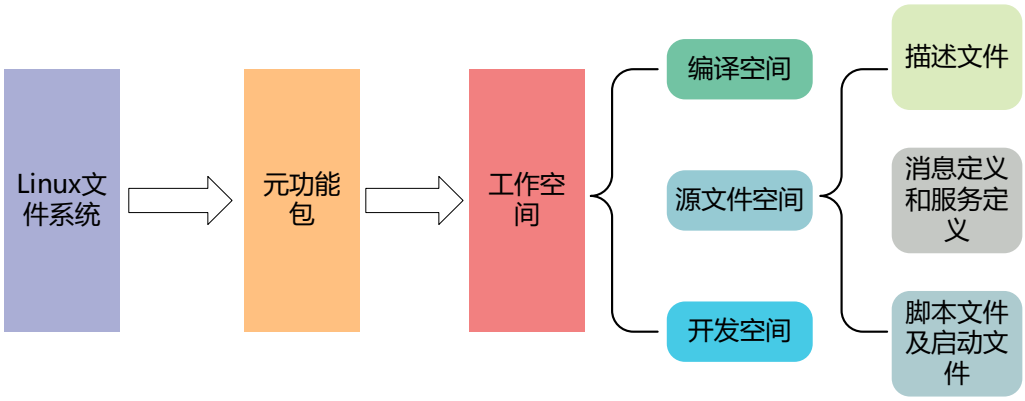


图 5-2 ROS 文件系统层

（3）开源社区

ROS 的开源社区通过开源代码、文档资源和活跃的开发者网络，促进了全球范围内智能车技术的快速发展与创新。ROS 开源社区由开发者、用户、贡献者、维护者等多种角色构成，这些角色各司其职，共同推动 ROS 的发展。开源社区支持的核心功能有：

开源代码共享：ROS 的所有核心代码和标准功能包都通过 GitHub、ROS 官网等平台公开，开发者可以自由下载、修改和分享。社区成员可以通过创建 Pull

Request 贡献代码，开发者之间协作改进功能包。

开源文档与学习资源：官方提供全面的文档，包括入门教程、API 参考、开发指南和系统设计原理。ROS Wiki 是社区维护的资源库，提供从基础安装到高级应用开发的教程和示例代码。

工具与依赖管理：ROS 社区维护了大量高质量的第三方功能包，这些包通过 rosdep 工具轻松集成到用户的项目中，同时官方提供了一站式的资源托管（如 ROS Index），方便开发者查找和使用开源功能包。

5.1.2 ROS 的特点

ROS 作为一个强大的开源机器人、智能车开发框架，具备许多显著特点，现已成为智能车研发领域的主流选择。以下是对 ROS 特点的简要介绍：

（1）模块化

众所周知，模块化设计是将每个功能进行独立的封装，只保留接口。模块化设计便于单独调试和维护，当某个模块需要升级或维修时，能从系统中独立拆分出来，而不影响其他模块的正常运行。在 ROS 中也采样用了模块化设计，将系统拆分为多个独立的功能模块，各个模块间功能解耦，可以单独的进行开发测试，同时用户可以灵活的添加设计新模块。

（2）分布式计算

分布式计算将复杂的计算任务分解为多个子任务，然后再多个计算机设备上分别运行单个子任务，最后在将运行结果进行汇总处理。ROS 处理的计算机任务大部分都是计算资源耗费大的任务，因此为了减少计算资源的损耗同时也为了维护系统的安全，ROS 采用了分布式计算的机制。ROS 通过网络协议实现多个节点在不同计算设备上的无缝通信，然后将计算任务分配到多个设备进行分布求解，以提高系统性能和运行效率，同时分布式特性允许其在不同硬件平台上灵活的部署节点，减轻了 ROS 部署硬件的难度。

（3）通信机制

ROS 是专为移动机器人和智能汽车设计的平台，为保障数据传输的实时性和鲁棒性，其内置了灵活高效的通信机制，能够适应多种复杂场景的数据交互需

求。基于发布-订阅模式的异步通信适用于传输图像、激光点云等海量数据的交互，有效降低系统的计算资源消耗；基于请求-响应模式的同步通信则以较高的实时性满足即时结果的需求。此外，扩展通信方式支持长时间任务的状态反馈与中断控制，进一步提升了系统的适应能力和操作效率。

（4）强大的仿真能力

ROS 集成了多种仿真工具，帮助开发者在虚拟环境中测试功能，减少了研究过程所需的硬件资源。集成的 Gazebo 仿真器支持物理建模和真实世界环境模拟，便于验证智能车的行为模式；RVIZ 可视化工具可以提供实时数据展示和交互功能，便于直观的理解算法效果。此外，ROS 的测试环境支持动态环境测试和多车辆协作测试。

5.2 搭建 ROS 仿真环境

Linux 是一种开源的类 Unix 操作系统，以其稳定性、安全性和高效性在多个领域广泛应用，故本章 ROS 环境选择 Ubuntu 18.04 作为底层操作系统。本章主要使用 Gazebo 仿真平台搭建复杂多变的测试环境和汽车的动力学模型，同时使用 Rviz 三维可视化平台直观的查看车辆的物理参数、实时的传感器数据、外界环境的变化等。下面介绍 Gazebo 仿真平台、Rviz 三维可视化平台以及测试环境的搭建过程。

5.2.1 Gazebo 仿真平台

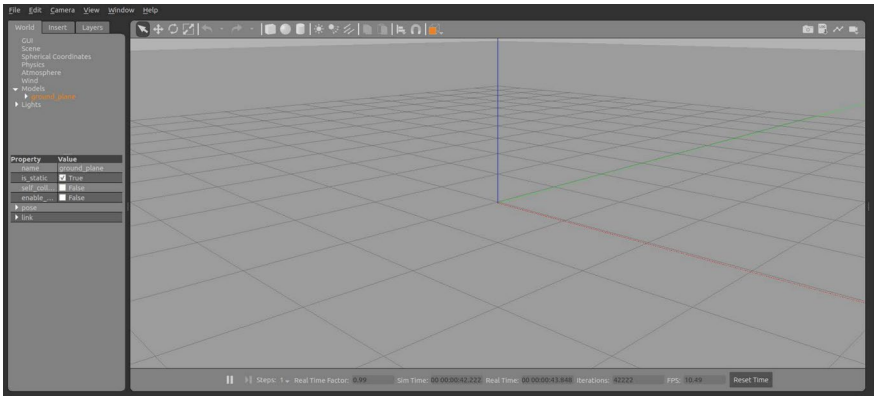


图 5-3 Gazebo 初始化界面

Gazebo 仿真平台是一款功能强大的开源仿真工具，在智能移动机器人及自

动驾驶系统研发领域具有广泛应用，Gazebo 平台如图 5-3 所示。该平台通过集成高精度物理引擎和多模态传感器仿真模块，能够实现对车辆运动学、动力学特性以及环境感知与决策控制行为的逼真模拟。其与 ROS 系统的深度兼容性支持，不仅实现了仿真环境与算法框架的无缝对接，还显著了提升了智能车的开发效率和测试可靠性。相比其他智能车仿真软件，其具有以下几个优点：

（1）真实的物理引擎支持。Gazebo 集成了多种高精度物理引擎，包括 ODE（Open Dynamics Engine）、Bullet、Simbody 和 DART 等，能够准确地模拟物体在不同环境下的运动。这些物理引擎支持刚体动力学、关节约束、摩擦模型、碰撞检测等功能，能够高度还原智能车在真实环境中的行为。此外，Gazebo 支持通过插件与 API 的方式扩展物理引擎功能，开发者可以根据需要增加自定义物理特性或模拟特定的环境条件，从而实现对复杂场景的精准建模。

（2）灵活的仿真模型描述。平台采用 URDF 和 SDF 作为模型定义语言，能够灵活地描述车辆的几何结构、动力学参数和运动学特性。该方法以 XML 文本文件为基础，对汽车模型进行质量、惯量、连杆、关节类型、传感器、材料等重要属性的定义，方便开发人员对其进行快速建模与修正。通过 Gazebo 模型库，开发者还可以直接使用或修改现有的高质量模型，进一步加快仿真场景的搭建过程。

（3）多样化的传感器仿真能力。Gazebo 提供了多种高精度的传感器模拟功能，支持激光雷达、深度相机、摄像头、惯性测量单元（IMU）、GPS 等多类型传感器的仿真。这些传感器能够生成仿真的感知数据，例如 3D 点云、图像、环境的深度信息和惯性数据等，为智能车的开发和测试提供了可靠的数据源。开发者可以灵活配置传感器的参数（如分辨率、采样频率、视场角、位置和朝向），从而适应不同的应用需求。同时，传感器数据可以通过 ROS 通信框架进行实时传输，使得仿真环境与实际的智能车系统无缝衔接。这样的能力不仅满足单个智能车的应用，还能支持多智能车场景下的协作感知与信息共享。

（4）强大的控制器支持。平台内置了多种类型的控制器，涵盖关节控制器、力控制器、速度控制器和轨迹控制器等，用于模拟不同智能车的运动控制需求。这些控制器通过插件形式实现，与 ROS 控制框架紧密结合，支持实时控制算法

的开发和验证。同时 Gazebo 允许用户开发自定义控制器插件，以满足特定任务的控制需求，例如在仿真中测试非线性控制算法、学习控制策略或多车协作控制。

5.2.2 Rviz 三维可视化平台

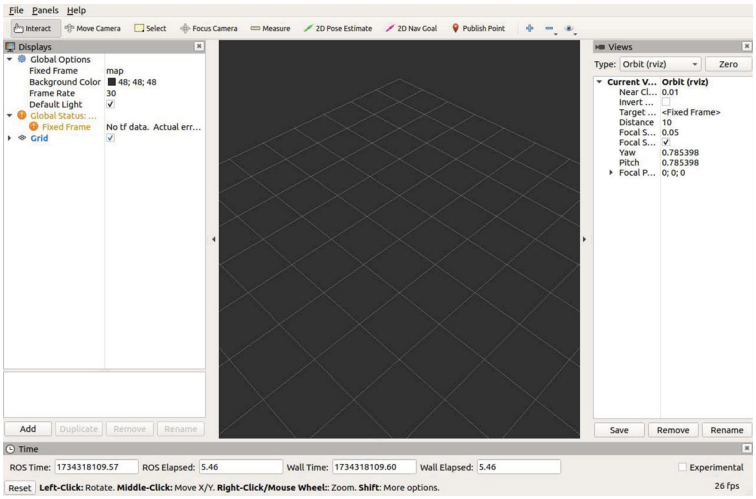


图 5-4 Rviz 初始化界面

Rviz 是一款基于 ROS 软件框架的三维可视化工具，广泛适用于智能车的研发。该工具支持通过 XML 格式定义智能车及其周围物体的尺寸、质量、位置、材质和关节等属性，并以图形化形式在界面中呈现。此外，Rviz 能够以可视化方式实时展示汽车的传感器数据、运动状态及周围环境的动态变化，为智能车的设计与分析提供直观的支持。Rviz 三维可视化平台如图 5-4 所示，具有以下显著特点：

（1）强大的三维重建能力。Rviz 的核心优势在于其卓越的三维重建功能。它能够将复杂的多维数据转化为直观的三维图像，使得数据的空间关系与结构特性得以清晰呈现。通过将传感器采集的环境信息（如点云、激光扫描等）整合到三维空间中，Rviz 能够模拟真实环境场景，从而为智能车的开发和调试提供更加生动的视觉反馈。

（2）高度可定制化。Rviz 提供了灵活的可视化元素和功能模块，使用户可以根据具体应用需求对显示内容进行高度定制。同时平台还允许用户添加自定义插件，以实现特定的功能扩展，如结合外部工具进行深度学习模型结果的实时可视化。通过支持多种显示方式（如栅格地图、路径轨迹、点云图等），Rviz 能满

足不同场景下的仿真与调试需求。这种高定制性使得 Rviz 不仅适合传统场景下的智能车研究，也能在非传统场景的车辆研究中发挥重要作用。

（3）交互式可视化功能。平台支持用户与显示内容的实时交互，从而显著提升了数据分析与系统调试的效率。用户可以通过拖拽、旋转和缩放等操作灵活调整视角，深入观察三维环境中某一区域的细节；还可以通过点击特定对象触发相应的功能，例如查看传感器输出的原始数据、分析车辆当前的位姿或观察特定路径的规划过程。此外，Rviz 的交互功能支持动态数据流的实时更新，开发者可以边操作边观察系统响应，快速定位问题并验证优化策略。这种互动方式不仅便于用户探索数据背后的关键信息，也为多车协作、复杂场景分析等任务提供了高效的调试手段。

（4）多数据源兼容性。Rviz 具备处理多种数据源的能力，展现出极高的灵活性与兼容性，能够满足不同场景下的数据可视化需求。其支持的输入数据类型包括本地文件（如点云文件、地图数据）、实时数据库，以及通过网络传输的在线数据流。这一特性使得 Rviz 能够高效整合多源数据，形成统一的三维可视化结果。

5.2.3 构建仿真测试环境

在 Gazebo 仿真平台中搭建简单静态测试环境，如图 5-5 所示。其中外围为用于封闭环境的墙体，内部灰色的为已知障碍物。在构建测试环境时，使用激光雷达和多种摄像机获取外界环境信息，以增强算法的适应性。

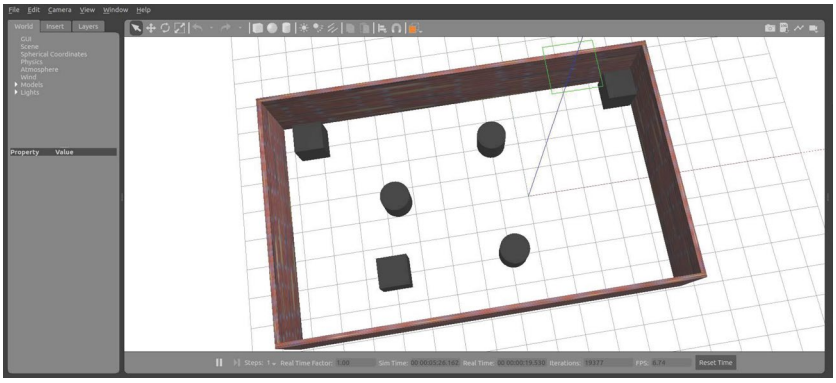


图 5-5 简单静态测试环境

在构建好仿真测试环境后，使用基于滤波 SLAM 框架的 Gmapping 算法建立

全局地图如图 5-6 所示。其中黑色为障碍物边界，灰色为未知区域，白色为可行区域。

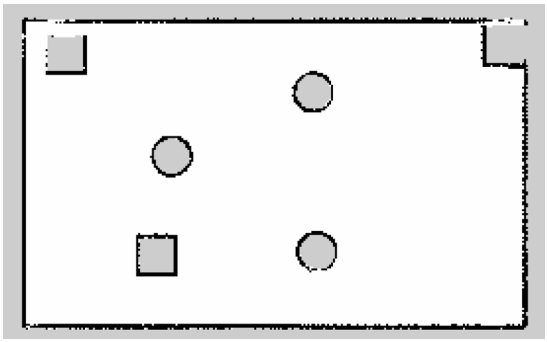


图 5-6 全局地图

为了验证算法在复杂动态环境下的性能，在已知环境的基础上添加未知障碍物和运动障碍物，复杂动态测试环境如图 5-7 所示。其中蓝色物体为未知障碍物，行人为运动障碍物。

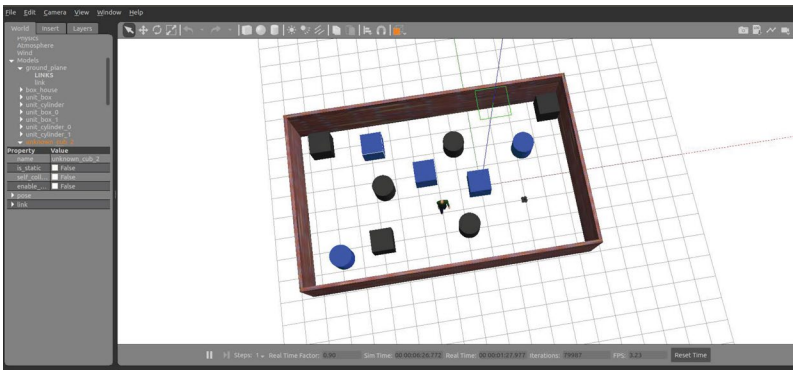


图 5-7 复杂动态测试环境

本章仿真测试所采用的智能车模型为 TurtleBot3-Waffle 模型，其结构如图 5-8 所示。TurtleBot3-Waffle 是一款小型化、低成本、完全可编程的智能车模型，基于 ROS 框架开发，广泛应用于科研及技术验证等领域。TurtleBot3-Waffle 模型凭借模块化设计和灵活性等特性已成为智能车研究的典型测试模型。

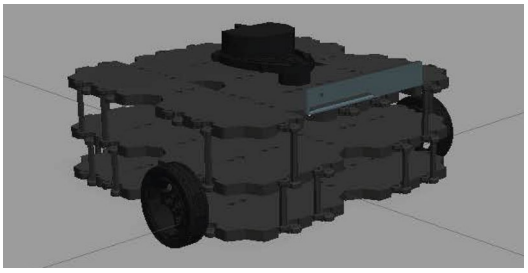


图 5-8 TurtleBot3-Waffle 模型

TurtleBot3-Waffle 模型的主要硬件组成包括底盘、电机、车轮、OpenCR 控制板、传感器和电池等模块。其动力系统采用高性能的 Dynamixel XM430 电机，结合两轮差速驱动方式，实现灵活的移动控制和轨迹跟踪能力；传感器系统集成了 3D RGB-D 相机和 2D 激光测距仪，用于获取环境深度信息和障碍物距离，为路径规划与避障算法提供精确的感知数据；姿态测量单元（IMU）配备三轴陀螺仪，能够实时测量车辆姿态参数，增强对运动状态的感知和控制。此外，TurtleBot3-Waffle 模型通过 OpenCR 控制板实现多传感器数据的高效融合与实时处理，保证了系统的协同运行。

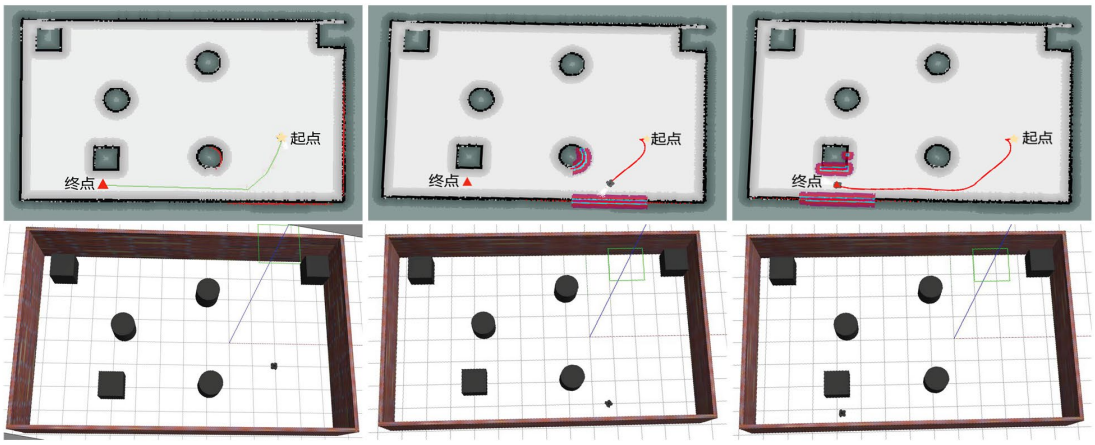
在性能方面，TurtleBot3-Waffle 模型最大平移速度为 0.26 m/s，最大角速度为 1.82 rad/s，能够胜任多种动态场景下的仿真测试需求。其模块化结构和可编程性使得用户可以根据具体实验需求对硬件进行扩展或对软件算法进行优化，从而在多样化的测试任务中表现出卓越的适应性和灵活性。

5.3 PSRRT*-BDWAHT 算法虚拟仿真实验与数据分析

5.3.1 简单静态环境下的可行性仿真测试

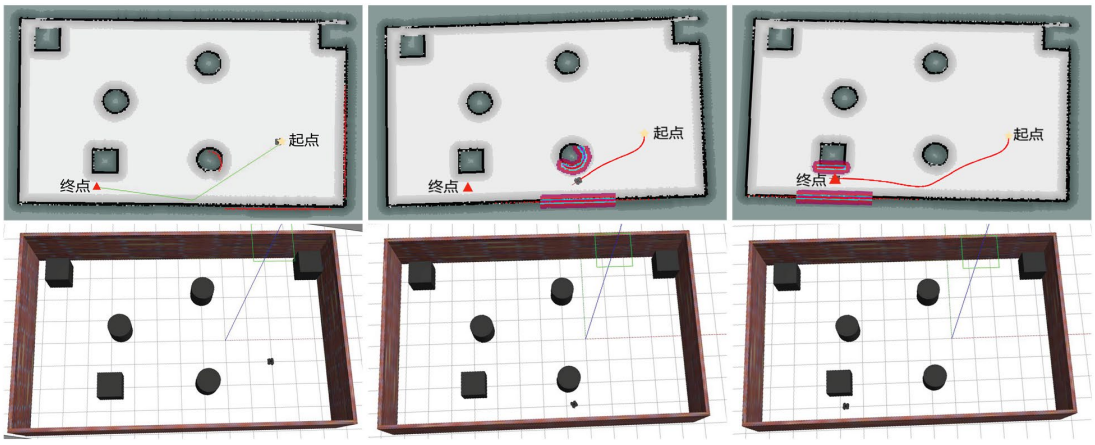
第四章的测试结果表明，在复杂动态环境中，PSRRT*-BDWAHT 算法规划的路径能够在保证安全性的前提下显著减少计算资源的消耗。然而，该测试未充分考虑车辆的物理约束以及外界环境的快速变化等因素。因此，本节将在 5.2.3 节构建的虚拟仿真环境中对所提出的算法进行测试，以进一步验证其可行性与适用性。

首先将算法配置为插件后，集成至 launch 启动文件中进行测试。运行该 launch 文件时，将自动启动 Gazebo 平台和 Rviz 平台。由于 launch 文件已完成 Gazebo 与 Rviz 的联动配置，因此用户可以通过 Rviz 界面中的“2D Pose Estimate”按钮设置车辆的初始位置，然后使用“2D Nav Goal”按钮来确定一个有方向的目标点。本节通过对比实验验证提出算法的性能，选用 PFRRT*-DWA 算法和 PSRRT*-BDWAHT 算法进行仿真测试。PFRRT*-DWA 算法的仿真结果如图 5-9 所示，PSRRT*-BDWAHT 算法的仿真结果如图 5-10 所示。



(a) 规划全局路径　(b) 对静态障碍物进行避障　(c) 完成路径规划

图 5-9 PFRRT*-DWA 算法静态环境仿真结果



(a) 规划全局路径　(b) 对静态障碍物进行避障　(c) 完成路径规划

图 5-10 PSRRT*-BDWAHT 算法静态环境仿真结果

表 5-1 简单环境下实验数据

算法	路径代价/cm	运行时间/s	平滑性	安全性
PFRRT*-DWA	50.44	24.34	平滑	安全
PSRRT*-BDWAHT	43.12	21.72	平滑	安全

图中上半部分为 Rviz 三维立体显示界面，下半部分为 Gazebo 仿真界面。在 Rviz 界面中，黄色五角星表示起点，红色三角形表示终点，白色区域表示可行区域，灰色区域为障碍物的膨胀层，彩色区域则代表局部代价层，绿色曲线为规划的全局路径，红色曲线为已行驶过的历史轨迹。

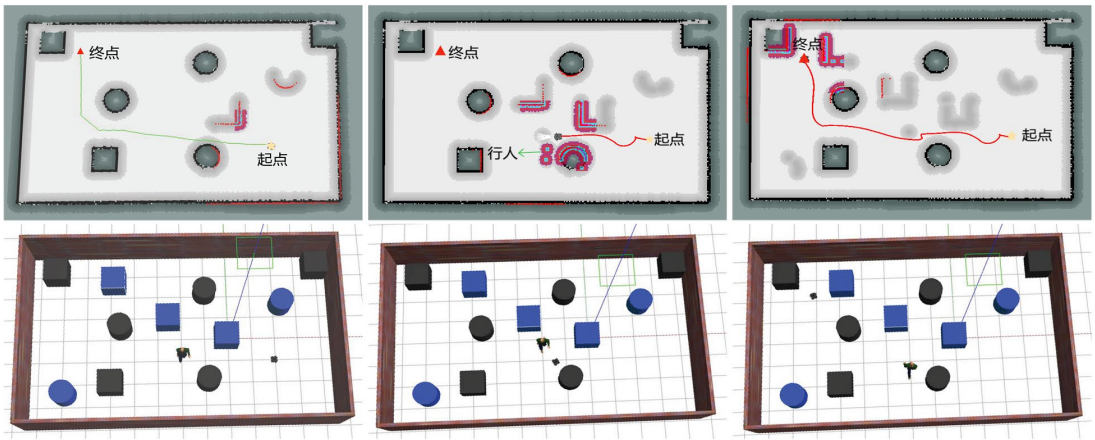
从图 5-9 (a)和图 5-10 (a) 所展示的全局路径规划结果可以看出，PSRRT*算法生成的全局路径具有更低的路径代价和更好的平滑性。图 5-9 (b)和图 5-10

(b) 中的避障过程表明，BDWAHT 算法能够有效避开障碍物，并严格跟踪全局路径。从整体路径规划结果来看，本文提出的算法生成的路径在平滑性方面表现更优，有助于提高车辆的行驶稳定性与安全性。从表 5-1 可以看出，在实际的简单静态环境中，PSRRT*-BDWAHT 算法相较于 PFRRT*-DWA 算法路径代价减少了 14.51%，运行时间减少了 10.76%，且行驶的路径更加平滑。仿真结果与实验数据进一步验证了所提算法在已知环境下的可行性和有效性。

5.3.2 复杂动态环境下的可行性仿真测试

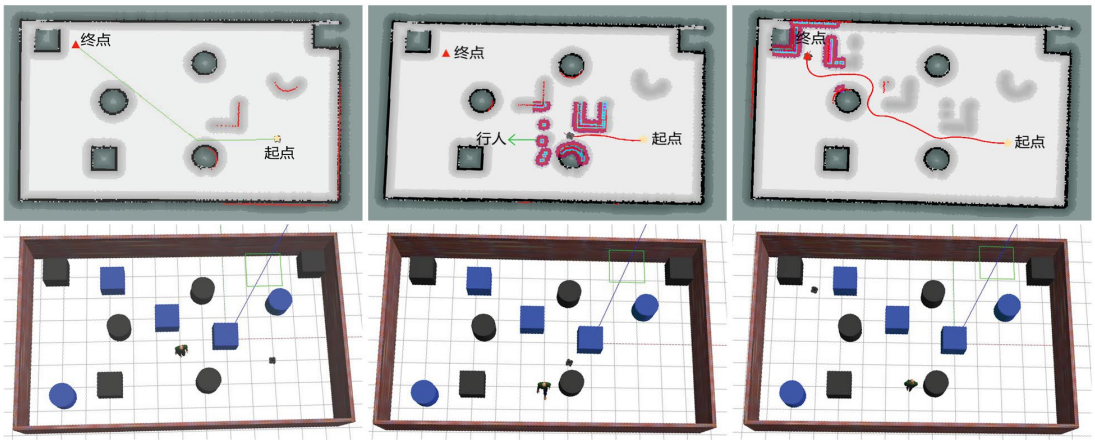
复杂动态测试环境如 5.2.3 节构建的环境所示，该环境包括未知障碍物和运动障碍物，整个环境复杂度高，规划难度大。本节同样使用 PFRRT*-DWA 算法和 PSRRT*-BDWAHT 算法进行仿真测试，以验证所提的 PSRRT*-BDWAHT 算法的性能。

PFRRT*-DWA 算法和 PSRRT*-BDWAHT 算法仿真结果如图 5-11 和图 5-12 所示。从表 5-2 可以看出 PSRRT*-BDWAHT 算法在路径代价上提升了 11.31%，运行时间减少了 18.01%。同时由图 5-11(c)和图 5-12(c)可以看出，在复杂动态环境下，PSRRT*-BDWAHT 算法生成的路径相较于 PFRRT*-DWA 算法，表现出更优的平滑性与更低的路径代价。这是由于 DWA 算法过于依赖全局路径且缺乏明确的目标导向性，从而导致规划路径缺乏安全性和平滑性。



(a) 规划全局路径　(b) 对动态障碍物进行避障　(c) 完成路径规划

图 5-11 PFRRT*-DWA 算法动态环境仿真结果



(a) 规划全局路径　(b) 对动态障碍物进行避障　(c) 完成路径规划

图 5-12 PSRRT*-BDWAHT 算法动态环境仿真结果

表 5-2 复杂动态环境下实验数据

算法	路径代价/cm	运行时间/s	平滑性	安全性
PFRRT*-DWA	83.22	47.71	不平滑	安全
PSRRT*-BDWAHT	73.81	39.12	平滑	安全

本节通过简单静态环境与复杂动态环境下的对比实验可以发现，所提出的 PSRRT*-BDWAHT 算法在路径代价、运行时间与路径平滑性方面均优于 PFRRT*-DWA 算法，展现了更好的整体性能。此外，仿真测试进一步验证了 PSRRT*-BDWAHT 算法在静态与动态环境中的适用性与可行性，证明了该算法在路径规划与避障任务中的实际应用价值。

5.4 基于 ZBOT3 智能车的 PSRRT*-BDWAHT 算法验证实验

为了验证提出的 PSRRT*-BDWAHT 算法的有效性和实用性，本节使用 Zbot3 智能车作为实车进行验证实验，并在室内搭建包含未知障碍物、移动障碍物及静态障碍物的测试环境进行实车的测试与分析。

5.4.1 Zbot3 智能车的组成结构

Zbot3 是一款基于 RK3568 处理器的 ROS 智能移动车，使用者可以通过 ROS 平台对算法进行验证分析。Zbot3 的组成结构如所示图 5-13 所示，本节主要介绍其中的主控、驱动和车轮结构。

图 5-13 Zbot3 组成结构

(1) 主控

Zbot3 智能车内置香橙派 Orange Pi 5B 开发板，瑞芯微 RK3588S 八核 64 位处理器、2.4GHz 主频；8GB+64GB 内存（LPDDR4，四通道）、支持 Wi-Fi 6 和蓝牙 5.0，支持 Android 12，Debian11、Ubuntu 22.04、Ubuntu 20.04 等操作系统。集成 ARM Mali-G610 MP4 GPU， 高达 6 TOPS 算力的 NPU。主控运行 ROS 系统，读取陀螺仪，激光雷达等传感器数据，通过运算，产生控制指令，并下发给辅助控制器或机械臂，由辅助控制器控制车运行。

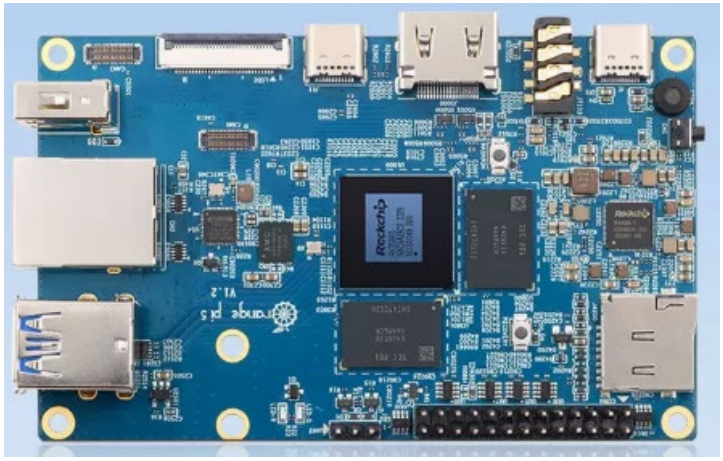


图 5-14 Zbot3 主控图

(2) 驱动电机

Zbot3 使用如所示图 5-15 的 4 个 MG513 直流减速电机，减速比为 30:1，单电机额定扭矩 1 kgf·cm，每个电机装有 GMR 编码器，分辨率为 500 PPR。电机使用 6-pin 的 XH2.54 接口（2.54 mm 间距），四个电机分别接 STM32 对应的电机接口。

图 5-15 MG513 直流减速电机

(3) 车轮结构

四个电机分别驱动 4 个直径 80 mm 的麦克纳姆轮，四个麦克纳姆轮分布如图 5-16 所示。四个麦克纳姆轮可以保证智能车实现全向移动，而无需改变车体朝向。这种轮组结构使得智能车具有极高的灵活性，能够轻松进行旋转、移动、侧向移动等复杂运动，适用于高机动性需求的场景。得益于麦克纳姆轮的物理特性，智能车在狭小空间或复杂环境中可以灵活避障，实现精准定位和任务执行。

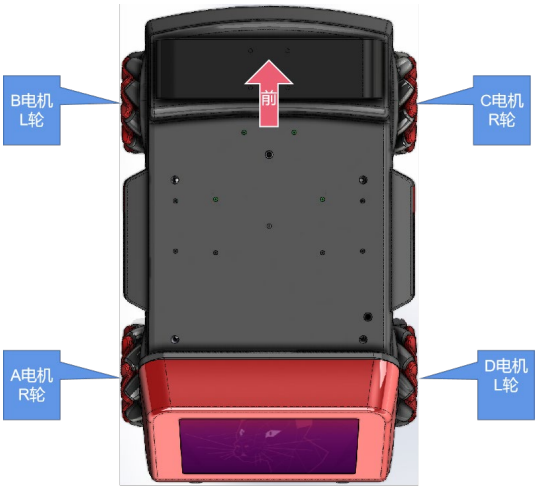


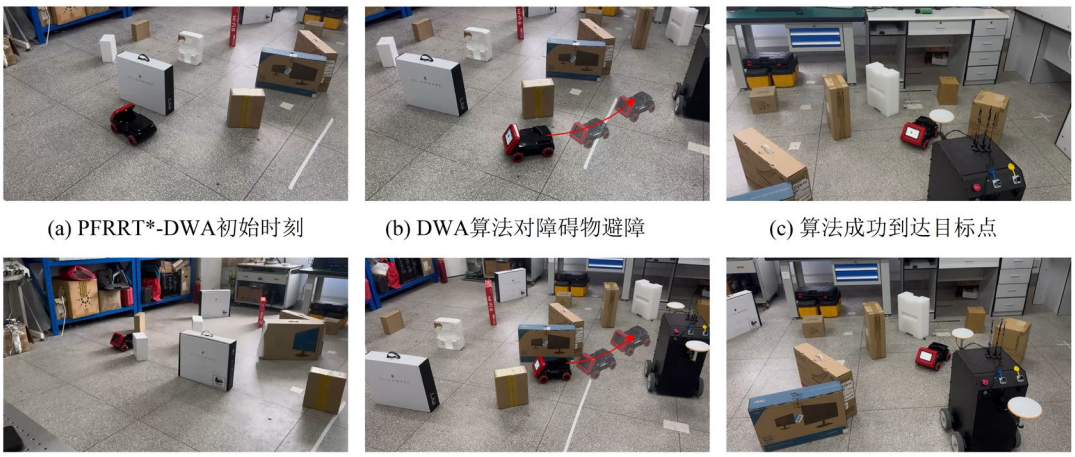
图 5-16 麦克纳姆轮分布图

5.4.2 基于真实环境的实车实验

为了测试真实环境中所提算法的性能，建立如所示的实验环境。在此环境下将使用 PFRRT*-DWA 算法和 PSRRT*-BDWAHT 算法进行对比实验，以验证所提算法的有效性和实用性。

根据 Zbot3 智能车的物理参数，设计了如图 5-17 所示的真实实验环境。真实地图中的星号代表起点，三角形代表终点，数字 1、2、3 和 4 代表未知障碍物，数字 5 的行人代表未知的移动障碍物。表 5-3 为真实环境下的实验数据。

图 5-17 真实测试环境



(a) PFRRT*-DWA初始时刻　(b) DWA算法对障碍物避障　(c) 算法成功到达目标点

(d) PSRRT*-BDWAHT初始时刻　(e) BDWAHT算法对障碍物避障　(f) 融合算法到达目标点

图 5-18 真实环境静态实验图

(a) PFRRT*-DWA初始时刻　(b) DWA算法对静态障碍物避障

(c) DWA算法对行人避障　(d) 成功到达目标点

图 5-19 PFRRT*-DWA 算法真实动态环境实验图

(a) PSRRT*-BDWAHT初始时刻　(b) BDWAHT算法对静态障碍物避障

(c) BDWAHT算法对行人避障　(d) 成功到达目标点

图 5-20 PSRRT*-BDWAHT 算法真实动态环境实验图

表 5-3 真实动态环境下实验数据

算法	路径代价/cm	运行时间/s	平滑性
PFRRT*-DWA	143.02	28.32	不平滑
PSRRT*-BDWAHT	130.53	25.34	平滑

图 5-18 为包含未知障碍物的真实静态环境实验测试结果，图中红色曲线为车辆的避障轨迹。从测试结果图可以看出，对于包含未知障碍物的静态环境，PFRRT*-DWA 算法和 PSRRT*-BDWAHT 算法均能流畅、快速的完成实车的路径规划任务。

图 5-19 为真实动态环境下 PFRRT*-DWA 算法的实验结果图，图 5-20 为真实动态环境下 PSRRT*-BDWAHT 算法的实验结果图，图中红色曲线为车辆的避障轨迹，黄色曲线为动态障碍物（行人）的运动轨迹，表 5-3 为真实动态环境下的实验数据。图 5-19(b)和图 5-20(b)分别为 DWA 算法和 BDWAHT 算法在静态障碍物避障过程的表现，二者均能实现平滑的避障效果。而在动态障碍物（行人）避障方面，图 5-19(c)和图 5-20(c)分别展现了 DWA 算法和 BDWAHT 算法的处理过程。DWA 算法因为计算复杂度高及与全局路径融合时路径波动，在对行人避障时会减慢速度。并且当距离行人较近时 DWA 算法可能停止行驶，等待行人走远。而 BDWAHT 算法通过使用双动态窗口模型和基于热传递的自适应窗口调

节机制，有效降低了算法计算复杂度，使其在复杂环境中能更迅速地调整路径，同时新的轨迹评价函数在融合全局路径时减少了路径波动，从而实现更加快速、顺滑的避障。从整个规格过程可以看出，尽管 PFRRT*-DWA 算法和 PSRRT*-BDWAHT 算法最终均能成功的到达目标点，但 PSRRT*-BDWAHT 算法在整个过程中表现得更为顺畅和快速。同时，从表 5-3 可以看出 PSRRT*-BDWAHT 算法相较于 PFRRT*-DWA 算法在路径代价上提升了 8.73%，运行时间减少了 10.52%。

真实世界下的实车实验表明，PSRRT*-BDWAHT 算法在复杂动态环境中仍能有效应对未知动态障碍物，保证较高的实时性、安全性与可行性，并提升了路径的全局最优性。

5.5 本章小结

在第 3 章全局路径规划和第 4 章局部路径规划的基础上，本章对 ROS 平台进行了简要介绍，并搭建了基于 ROS 的虚拟仿真测试环境和真实世界测试环境。两种不同的虚拟仿真测试和真实世界的实车实验验证了提出的 PSRRT*-BDWAHT 算法在物理世界的有效性和可行性。

第6章 总结与展望

6.1 工作总结

随着科技的迅猛发展，智能车在各行各业越发重要，而路径规划技术作为智能车的关键技术自然也受到研究者的重点关注。本文针对复杂环境下的路径规划问题，提出一种 PSRRT*-BDWAHT 算法，以提升路径的规划率。本文通过仿真实验和实车实验验证了所提算法的有效性和可行性。本文的主要工作总结如下：

(1) 通过阅读论文、调研行业发展等方法对智能车的路径规划发展背景、研究现状进行深入的了解，并对当前主流的路径规划算法进行了总结与分析，了解各类路径规划算法的利弊以及不同规划算法间的差异性，为解决当前路径规划在复杂动态环境中存在的实时性差、安全性低等问题提供理论基础。

(2) 在路径规划任务中，全局路径提供一个整体的行进方向，并完成静态避障任务。但是传统的全局路径规划算法存在计算复杂度大、采样点导向性差、路径平滑性低等问题，为此本文提出一种 PSRRT*算法。该算法建立精确采样模型增强采样点的目标导向性，以加快算法收敛速度；利用提出的自适应节点扩展模型生成随机树，以减少计算复杂度；最后对规划的初始全局路径进行剪枝平滑处理，以增强路径的平滑性。不同环境下的仿真测试验证了 PSRRT*算法的有效性。

(3) 考虑面对动态环境或未知环境时全局路径规划算法无法实时的调整路径，无法保证路径的安全性，本文提出一种 BDWAHT 算法。首先采用双动态窗口模型减少预测轨迹时的障碍物信息，提高了算法的实时性；其次使用热传递方法自适应的调整窗口尺寸大小，增强了算法的泛用性。仿真对比实验验证了 BDWAHT 算法的有效性。

(4) 为了验证提出的 PSRRT*-BDWAHT 融合算法在物理世界的有效性和可行性，本文搭建了基于 ROS 平台的虚拟仿真测试和实车实验。在基于 ROS 平台的虚拟仿真测试中，使用 Gazebo 仿真平台和 Rviz 三维可视化平台进行联合仿真。在基于 ROS 平台的实车测试中，利用 Zbot3 智能车实现了真实场景中的自主导航。虚拟仿真测试和实车实验验证了本文提出的融合算法的有效性、合理性。

6.2 工作展望

本文对国内外学者在路径规划领域的研究成果进行了分析与总结，提出了PSRRT*全局路径规划算法与 BDWAHT 局部路径规划算法。虽然本文提出的算法在复杂环境下的路径规划任务上取得了一定的成效，但是由于客观原因与时间限制导致本文仍存在不足之处，未来将在此基础上进一步研究。

(1) 本文只考虑了复杂环境下单个智能车的路径规划问题，但是在实际应用场景中一般存在多个智能车协同工作。未来的工作将基于多智能体系统理论，考虑车辆间的通信机制、任务分配与协同决策，研究多车协同路径规划与冲突解决策略。

(2) 在设计复杂环境时，只考虑了未知障碍物以及有运动轨迹的动态障碍物，没有考虑突然出现的障碍物，对于一些场景的路径规划具有一定的限制。未来将引入更丰富的障碍物类型，通过构建更加动态、多变和不确定性的环境模型，进一步验证规划算法的鲁棒性与实时响应能力。此外，还将研究异常事件检测与快速应对机制，提升系统在极端工况下的稳定性和安全性。

(3) 在基于 ROS 平台的虚拟仿真和实车实验中主要使用的是激光雷达获取环境信息，对环境信息的采集过于单一，无法捕捉更多的信息。未来将拓展实车实验的环境复杂度，验证算法在实际动态复杂环境中的适应性与鲁棒性。

参考文献

[1] 邓三鹏, 张香玲, 王凯, 等. 具身智能机器人关键技术及发展趋势研究[J]. 装备制造技术, 2024, (06):2-10.

[2] Ma R, Hou Y L, Wang S R. Combatting cancer with tiny intelligent warriors: Stimuli-responsivemicro/nanorobots in chemotherapeutic delivery[J]. Science China Materials, 2024, 67(8):2469-2476.

[3] 梅紫琦, 金胜姬, 李玮彤, 等. 智能机器人在护理健康教育领域中应用的范围综述[J]. 护理学报, 2024, 31(07):57-62.

[4] 王伟, 王勇, 张晔, 等. 智能消毒机器人移动平台导航系统研究[J]. 机械设计与制造, 2024, 11:346-350.

[5] 王湉, 范峻铭, 郑湃.基于大语言模型的人机交互移动检测机器人导航方法[J]. 计算机集成制造系统, 2024, 30(05):1587-1594.

[6] 袁瑞萍, 邹顺洁, 潘路可, 等. 基于移动机器人的拣选系统货架动态储位分配研究[J]. 系统科学与数学, 2024, 44(03):780-791.

[7] Zhang Q, Zhao J, Pan L, et al. Optimal path planning for mobile robots in complex environments based on the gray wolf algorithm and self-powered sensors[J]. IEEE Sensor Journal, 2023, 23(18):20756-20765.

[8] Badue C, Guidolini R, Carneiro R V, et al. Self-driving cars: a survey[J]. Expert Systems with Applications, 2021, 165:113816.

[9] 兰沣卜, 赵文博, 朱凯,等. 基于具身智能的移动操作机器人系统发展研究[J]. 中国工程科学, 2024, 26(01):139-148.

[10] Huang Z X, Li X P, Wei Z, et al. A stable control method for planar robot with underactuated constraints via motion planning and intelligent optimization[J]. Measurement & Control, 2023, 56(9):1826-1834.

[11] Liu M S, Wan Y, Lewis F L, et al. A three-level game-theoretic decision-making framework for autonomous vehicles[J]. IEEE Transactions on Intelligent Transportation Systems, 2022, 23(11):20298-20308.

[12] 宋荆洲, 宫兴龙, 段嘉辰, 等. 可跳跃移动机器人机构设计与跳跃过程控制研究综述[J]. 机械工程学报, 2024, 60(15):1-17.

[13] 张昌昕, 张兴龙, 徐昕,等. 安全强化学习及其在机器人系统中的应用综述[J].

控制理论与应用, 2023, 40(12):2090-2103.

[14] Tagliavini L, Colucci G, Botta A, et al. Wheeled mobile robots: state of the art overview and kinematic comparison among three omnidirectional locomotion strategies[J]. Journal of Intelligent & Robotic Systems, 2022, 106:57.

[15] Liang X, Liu C, Song X, et al. Research on improved artificial potential field approach in local path planning for mobile robot[J]. Computer simulation, 2018, 35(4):291-361.

[16] 王乐, 齐尧, 何滨兵, 等. 机器人自主探索算法综述[J]. 计算机应用, 2023, 43(S1):314-322.

[17] SAE International. J3216 Taxonomy and definitions for terms related to co opera-tive driving automation for on-road motor vehicles[S/OL]. https://www.sae.org/standards/content/j3216/.

[18] Zhao J Y, Zhao W Z, Deng B, et al. Autonomous driving system: a comprehensive survey[J]. Expert Systems with Applications, 2024, 242:122836.

[19] Zhang Y X, Carballo A, Yang H T, et al. Perception and sensing for autonomous vehicles under adverse weather conditions: a survey[J]. ISPRS Journal of Photogrammetry and Remote Sensing, 2023, 196:146-177.

[20] Tzafestas S G. Mobile robot control and navigation: a global overview[J]. Journal of Intelligent & Robotic Systems, 2018, 91(1):35-38.

[21] 毛建旭, 贺振宇, 王耀南, 等. 电力巡检机器人路径规划技术及应用综述[J]. 控制与决策, 2023, 38(11):3009-3024.

[22] 高春艳, 陶渊, 吕晓玲, 等. 非结构化环境下巡检机器人环境感知技术研究综述[J]. 传感器与微系统, 2023, 42(04):10-18.

[23] Peter E H, Nils J N, Bertram R, et al. A Formal Basis for the Heuristic Determination of Minimum Cost Paths[J]. IEEE Transactions on Systems Science and Cybernetics, 1968, 4(2):100-107.

[24] Huang C S, Zhao Y P, Zhang M J, et al. APSO: An A∗-PSO hybrid algorithm for mobile robot path planning[J]. IEEE Access, 2023, 11:43238-43256.

[25] Ou J L, Seong H H, Song G, et al. Hybrid path planning based on adaptive visibility graph initialization and edge computing for mobile robots[J]. Engineering Applications of Artificial Intelligence, 2023, 126(D):107-110.

[26] Qureshi A H, Ayaz Y. Potential functions based sampling heuristic for optimal

path planning[J]. Autonomous Robots, 2015, 40:1079-1093.

[27] Fan J M, Chen X, Xiao L, et al. UAV trajectory planning based on bi-directional APF-RRT* algorithm with goal-biased[J]. Expert Systems with Applications, 2023, 213(C):119137-119148.

[28] Zhang J C, An Y Q, Cao S B, et al. UAV trajectory planning for complex open storage environments based on an improved RRT algorithm[J]. IEEE Access, 2023, 11:23189-23204.

[29] 李宗刚, 韩森, 陈引娟, 等. 基于角度搜索和深度 Q 网络的移动机器人路径规划算法[J]. 兵工学报, 2025, 46(02):32-46.

[30] 张磊, 母亚双, 潘泉. 基于改进深度双 Q 网络的移动机器人路径规划算法[J]. 信息与控制, 2024, 53(03):365-376.

[31] He Z B, Liu C G, Chu X M, et al. Dynamic anti-collision A-star algorithm for multi-ship encounter situations[J]. Applied Ocean Research, 2022, 118:102995.

[32] Zhou L T, Wu N P, Chen H, et al. RRT*-Fuzzy Dynamic Window Approach (RRT*-FDWA) for collision-free path planning[J]. Advances in Robot Path Planning, 2023, 13(9):5234-5250.

[33] Kim H, Yun S J, Choi Y H, et al. Improved dynamic window approach with ellipse equations for autonomous navigation of unmanned surface vehicle[J]. Journal of Institute of Control, Robotics and Systems, 2020, 26(8):624-629.

[34] Wu J F, Ma X H, T G, et al. An improved timed elastic Band(TEB) algorithm of autonomous ground vehicle(AGV) in complex environment[J]. Sensors, 2021, 21(24): 8312-8312.

[35] Wang Z, Zhang S, Feng X, et al. Autonomous underwater vehicle path planning based on actor-multi-critic reinforcement learning[J]. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 2021, 235(10): 1787-1796.

[36] Yin Y, Chen Z, Liu G, et al. A Mapless Local Path Planning Approach Using Deep Reinforcement Learning Framework[J]. Sensors, 2023, 23(4): 2036.

[37] 徐建华, 邵康康, 王佳惠, 等. 基于改进强化学习的移动机器人动态避障方法[J]. 中国惯性技术学报, 2023, 31(1): 92-99.

[38] 鲁帆. 基于移动机器人的路径规划算法研究[D]. 安徽工程大学, 2023.

[39] Zhao Z Y, Liu H O, Chen H Y, et al. Kinematics-aware model predictive control

for autonomous high-speed tracked vehicles under the off-road conditions[J]. Mechanical Systems and Signal Processing, 2019, 123:333-350.

[40] Qin D, Dong H, Sun S, et al. Model-Driven cooperative path planning for dynamic target searching of unmanned unterwater vehicle formation[J]. Marine Science and Engineering, 2024, 12:2094.

[41] He Z B, Liu C G, Chu X M, et al. A topology based path planning algorithm for inland waterway network: case study of Jiangsu Canal Network[C]// /Proceedings of the 7th International Conference on Transportation Information and Safety. IEEE, 2023:1767-1773.

[42] Zhang R, Guo H, et al. Intelligent path planning by an improved RRT algorithm with dual grid map[J]. Alexandria Engineering Journal, 2024, 88:91-104.

[43] Reda M, Onsy A, Haikal A Y, et al. Path planning algorithms in the autonomous driving system: A comprehensive review[J]. Robotics and Autonomous Systems, 2024, 174:104630.

[44] Miyombo E, Liu Y K, et al. Optimal path planning in a real-world radioactive environment: A comparative study of A-star and Dijkstra algorithms[J]. Nuclear Engineering and Design, 2024, 420:113039.

[45] Li H L, Qian L X, Hong M, et al. Effective anti-submarine decision support system based on heuristic rank-based Dijkstra and adaptive threshold partitioning mechanism[J]. Applied Soft Computing, 2024, 161:111718.

[46] Liao T J, Chen F, Wu Y T, et al. Research on path planning with the Integration of Adaptive A-Star Algorithm and Improved Dynamic Window Approach[J]. Electronics, 2024, 13(2):455.

[47] Zuo S T, Mao Z L, Fan C G, et al. Dynamic planning of crowd evacuation path for metro station based on Dynamic Avoid Smoke A-Star algorithm[J]. Tunnelling and Underground Space Technology, 2024, 154:106145.

[48] Guo T, Sun Y Q, Liu Y, et al. An automated guided vehicle path planning algorithm based on improved A* and dynamic window approach fusion[J]. Applied Sciences, 2023, 13(18):10326.

[49] 薛光辉, 刘爽, 王梓杰, 等. 基于改进概率路线图算法的煤矿机器人路径规划方法[J]. 工矿自动化, 2023, 49(06):175-181.

[50] Zheng X C, Chao J J, Zhang B, et al. Path planning of PRM based on artificial potential field in radiation environments[J]. Annals of Nuclear Energy, 2024,

208:110776.

[51] Ye J T, Hao L N, Cheng H T, et al. A novel global path planning method for robot based on dual-source light continuous reflection[J]. ISA Transactions, 2024, 150:15-29.

[52] Zhong H G, Cong M, et al. HB-RRT: A path planning algorithm for mobile robots using Halton sequence-based rapidly-exploring random tree[J]. Engineering Applications of Artificial Intelligence, 2024, 133(E):108362.

[53] Cui X N, Wang C Q, et al. More Quickly-RRT*: Improved Quick Rapidly-exploring Random Tree Star algorithm based on optimized sampling point with better initial solution and convergence rate[J]. Engineering Applications of Artificial Intelligence, 2024, 133(C):108246.

[54] 刘志华, 张冉, 郝梦男, 等. 基于改进 T 分布烟花-粒子群算法的 AUV 全局路径规划[J]. 电子学报, 2024, 52(09):3123-3134.

[55] Sun R, Yang T. Hybrid parameter-based PSO flexible needle percutaneous puncture path planning [J]. Journal of Supercomputing, 2024, 80(4):5408-5427.

[56] Li J, Li J, Fang H J, et al. Dynamic energy-efficient path planning for electric vehicles using an enhanced Ant Colony Algorithm[J]. Tehnički Vjesnik Technical Gazette, 2024, 31(2):434-441.

[57] Cui J G, Wu L, Huang X D, et al. Multi-strategy adaptable ant colony optimization algorithm and its application in robot path planning[J]. Knowledge-Based Systems, 2024, 288.

[58] Mishra P, Jain U, Choudhury S, et al. Footstep planning of humanoid robot in ROS environment using Generative Adversarial Networks (GANs) deep learning[J]. Robotics and Autonomous Systems, 2022, 158:104269.

[59] Tang W B, Zhou Y, Sun H Y, et al. GAN-Based robust motion planning for mobile robots against localization attacks[J]. IEEE Robotics and Automation Letters, 2023, 8(3):1603-1610.

[60] Li J X, Chen Y T, Zhao X N, et al. An improved DQN path planning algorithm[J]. The Journal of Supercomputing, 2021, 78:616-639.

[61] Gu Y W, Zhu Z T, Lv J D, et al. DM-DQN: Dueling Munchausen deep Q network for robot path planning[J]. Complex & Intelligent Systems, 2023, 9(4):4287-4300.

[62] Fox D, Burgard W, Thrun S. The dynamic window approach to collision

avoidance[J]. IEEE Robotics & Automation Magazine, 1997, 4(1):23-33.

[63] Wang J Y, Luo Y H, Tan X J, et al. Path planning for Automatic Guided Vehicles (AGVs) fusing MH-RRT with improved TEB[J]. Actuators, 2022, 10(12):314-332.

[64] Liu Z Y, Zhu D Q, Liu C X, et al. A novel path planning algorithm of AUV with Model Predictive Control[J]. International Journal of Robotics & Automation, 2022, 37(6): 460-467.

[65] Ganesan S, Ramalingam B, Mohan R E, et al. A hybrid sampling-based RRT* path planning algorithm for autonomous mobile robot navigation[J]. Expert Systems with Applications, 2024, 258: 125206.

[66] Yu J X, Chen C, Arab A, et al. RDT-RRT: Real-time double-tree rapidly-exploring random tree path planning for autonomous vehicles[J]. Expert Systems with Applications, 2024, 240:122510.

[67] Gammell J D, Srinivasa S S, Barfoot T D. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic[C]// Proceedings of 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2014, 2997-3004.

[68] Fan J M, Chen X, Yu W, et al. UAV trajectory planning in cluttered environments based on PF-RRT* algorithm with goal-biased strategy[J]. Engineering Applications of Artificial Intelligence, 2022, 114:105182.

[69] Liu Y J, Wang C, Wu H, et al. Mobile robot path planning based on kinematically constrained A-star algorithm and DWA fusion algorithm[J]. Mathematics, 2023, 11(21):4552-4571.

攻读硕士期间发表文章

[1] Han C, **Yu Z Y**, Shi X, et al. Dynamic Path Planning for Rapidly Expanding Autonomous Vehicles in Complex Environments[J]. IEEE Sensor Journal, 2025, 25(1): 1216 – 1229.

致谢

当毕业论文落下最后一笔，三载春秋的研究生求学之路也即将抵达终点。临窗回望，这一程跋涉既有登攀学术高峰的艰辛，更满载着恩师亲友馈赠的温暖。值此论文付梓之际，谨以拳拳之心向所有助我前行的引路人与同行者致以最深沉的谢忱。

首先要深深感谢我的导师韩超教授。三年来，您用严谨的治学态度为我树立学术标杆，以春风化雨般的耐心指引我成长。在论文写作过程中，从中文初稿到英文翻译，您总是不厌其烦地逐字批改，通过不断的打磨与指导，最终帮助我成功发表了文章。韩老师的辛勤付出让我在学术道路上不断成长，感激之情无以言表。

其次，我要感谢我的同学们。在毕业之际，我衷心祝愿你们拥有幸福的明天和美好的未来，愿我们的友谊地久天长。

同时，我要特别感谢我的父母。感谢你们多年来的辛勤付出，为我创造了良好的生活环境。你们的陪伴和支持是我顺利完成研究生学业的最大动力。无论未来如何，你们的爱永远是我前行的力量源泉。

最后，我要向在百忙之中抽出时间审阅、评议我的论文并参与答辩的各位老师表示诚挚的感谢。你们的宝贵意见和建议对我的研究工作起到了至关重要的作用，我将铭记于心。

这段求学之旅的结束，也是新征程的开始。我将怀着感恩之心，继续前行，不负众望。

尚德教学　唯实惟新