分类号：TP242　　学校代码：10363
密　级：公　开　　学　　号：2220320140

安徽工程大学
Anhui Polytechnic University

硕士学位论文

题目　智能移动充电机器人路径规划研究

论文作者　胡浩然
校内导师　方杰
校外导师　李权

专业学位类别　电子信息
研究方向（领域）　智能信息处理与应用

论文提交日期：2025 年 5 月 15 日

分类号：TP242　　单位代码：10363

密　级：公 开　　学　　号：2220320140

题　　目　智能移动充电机器人路径规划研究

题　　目　Research on Path Planning for Intelligent Mobile Charging Robots

学生姓名：　胡浩然

校内导师：　方杰（教授）

校外导师：　李权

专业学位类别：　电子信息

研究方向（领域）：　智能信息处理及应用

论文答辩日期：2025 年 5 月 10 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 15 日

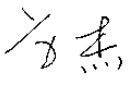
安徽工程大学学位论文版权使用授权书

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 15 日　　日期：2025 年 5 月 15 日

智能移动无人驾驶充电机器人路径规划研究

摘 要

在新能源汽车技术持续发展的今天，移动充电机器人的需求日益增长，极大促进了自动驾驶技术的研究与发展。而如今存在的充电机器人在激光雷达点云配准和路径规划算法方面仍处于不成熟阶段，例如在复杂环境中的算法稳定性不高、算法本身不具备安全保障等。针对以上问题，本文对激光雷达点云配准算法和已知地图下的全局路径规划算法进行了相关研究。研究结果表明改进后的算法有效提高了点云配准质量和路劲规划质量，并在自动驾驶测试中取得了不错的效果。

本文的主要内容如下：

（1）对充电机器人的实际作业场景进行分析，并基于Ubuntu20.04 系统下的 ROS 环境进行仿真环境与模型的搭建。由GAZEBO 自带的模型菜单创建物理仿真环境并使用 SOLIDWORKS 设计机器人结构模型并导出 URDF 格式到 ROS 中，为后续控制机器人在仿真环境中移动采集数据奠定基础。

（2）针对传统迭代最近点算法(Iterative Closest Point, ICP)受点云初始位置影响较高、噪声干扰较大、配准效率较低等问题，首先对采集到的点云数据使用内蕴形状特征签名算法（Intrinsic Shape Signature, ISS）提取特征点，并进行快速点特征直方图描述，通过随机样本一致算法（Random Sample Consensus, RANSAC）剔除错误点对完成粗

配准，获得较为接近的两片初始点云；在 ICP 精配准确定两片点云对应关系前首先对离散点进行判定，设置搜索半径阈值 r 获取最佳点云信息，借助 KDtree 结构加速对应点对的搜索，最后完成点云精配准。仿真结果表明改进后的算法在对应点搜寻数量上分别减少了 15.62%、15.04%，总体配准质量上优于其他算法。

（3）针对充电机器人在进行路径规划时算法不具备安全保障、冗余节点较多、计算量较大等问题，在 A*算法的基础上进行改进：首先优化常规 A*算法的启发式函数，使估计代价接近于真实值并指引路径趋向于终点；其次对搜索节点进行正方向判断，减少冗余节点数量；向对角线节点的临节点进行障碍物判定，为小车行驶提供安全保障；最后通过 B 样条曲线对生成的路径做平滑处理。仿真结果表明改进后的 A*算法，目标点无论是在空旷处还是复杂环境中，都能很好的规划出一条可行的安全路径。

（4）基于 ROS 的智能导航系统进行自动驾驶的实际测试。首先确定机器人的移动方案，包括手柄遥控和自主导航两种控制模式，使用激光雷达 SLAM 算法构建实验场地地图，其次考虑到机器人的行驶安全，在测试中采用全局和局部规划器共同制导框架：改进 A*算法规划全局路径后再由动态窗口法（Dynamic Window Approach, DWA）调整局部轨迹。测试结果显示，机器人基于已构建地图规划出的路径实时避开障碍物并安全抵达终点，实现了小车自主导航功能。

关键词：充电机器人，ROS，点云配准，路径规划，自主导航

RESEARCH ON PATH PLANNING FOR INTELLIGENT MOBILE AUTONOMOUS ROBOTS

ABSTRACT

With the continuous development of new energy technologies, the demand for mobile charging robots has significantly increased, greatly promoting research and advancement in autonomous driving technology. However, existing charging robots still face challenges in LiDAR point cloud registration and path planning algorithms, such as unstable algorithm performance in complex environments and insufficient safety guarantees. To address these issues, this paper conducts research on LiDAR point cloud registration algorithms and global path planning algorithms based on known maps. The results demonstrate that the improved algorithms effectively enhance the quality of point cloud registration and path planning, achieving promising outcomes in autonomous driving tests for robots.

The main contributions of this paper are as follows:

(1) The practical working scenarios of the charging robot were analyzed, and a simulation environment along with models was constructed using the Robot Operating System (ROS) framework on Ubuntu 20.04. A physical simulation environment was created using

GAZEBO's built-in model library, while the robot's structural model was designed in SOLIDWORKS and exported in URDF (Unified Robot Description Format) to ROS. This groundwork enabled subsequent control of the robot's movement and data collection within the simulation environment.

(2) To address the limitations of the traditional Iterative Closest Point (ICP) algorithm—such as high sensitivity to initial point cloud positions, susceptibility to noise, and low registration efficiency—an enhanced approach was implemented. First, the collected point cloud data was processed using the Intrinsic Shape Signature (ISS) algorithm to extract feature points, followed by Fast Point Feature Histogram (FPFH) feature description. Mismatched point pairs were eliminated via the Random Sample Consensus (RANSAC) algorithm to achieve coarse registration, producing two closely aligned initial point clouds. Before establishing correspondence in the ICP fine registration stage, discrete points were evaluated, a search radius threshold r was set to optimize point cloud information retrieval, and a KDtree structure was employed to accelerate correspondence pair search, ultimately completing precise point cloud registration. Simulation results demonstrate that the improved algorithm reduces the number of correspondence searches by 15.62% and 15.04%, respectively, while outperforming other algorithms in overall registration quality.

(3) To address the challenges of insufficient safety guarantees, excessive redundant nodes, and high computational load in path planning for charging robots, an improved A* algorithm was developed. The conventional heuristic function of the A* algorithm was optimized to make the estimated cost closer to the true value, thereby guiding the path toward the target. A forward-direction judgment mechanism was introduced during node searching to reduce redundant nodes. Obstacle detection was implemented for adjacent nodes of diagonal paths to enhance driving safety. Finally, B-spline curves were applied to smooth the generated paths. Simulation results demonstrate that the improved A* algorithm successfully plans feasible and safe paths to targets in both open and complex environments.

(4) Practical autonomous navigation tests were conducted using a ROS-based intelligent navigation system. First, the robot's mobility scheme was defined, including two control modes: joystick remote control and autonomous navigation. A LiDAR SLAM algorithm was utilized to construct a map of the experimental environment. To ensure driving safety, a hybrid guidance framework combining global and local planners was adopted during testing: the improved A* algorithm planned global paths, followed by local trajectory adjustments using the Dynamic Window Approach (DWA). Test results demonstrated that the robot dynamically avoided obstacles and safely reached targets based on the planned paths

from the constructed map, successfully achieving autonomous navigation functionality.

Hu Haoran (Robot Control)

Supervised by Fang Jie

KEY WORDS: Charging robot, ROS, Point cloud registration, Path planning, Autonomous navigation

目 录

第 1 章 绪论

1.1 研究背景及意义

自 1920 年捷克作家卡雷尔·恰佩克提出“robot”概念以来[1]，机器人技术经历了跨越式的发展。根据国际标准化组织的定义，“机器人是具有一定程度的自主能力，可在其环境内运动以执行其预期任务的可编程执行机构”[2]。当前机器人已渗透至工业制造、医疗护理等场景，近些年中国机器人产业保持持续高速增长[3]。随着全球对可持续能源的关注度增加，新能源技术和智能设备快速崛起。移动充电机器人也在全球范围内迅速发展，需求量不断增长，智能移动充电机器人也需要适应不同的应用场景和环境挑战。由人工智能与机器学习算法技术驱动，机器人自动驾驶技术更新了传统的移动模式，通过深度集成自主感知、决策与控制系统，使机器人向高度自主驾驶逐步演进[4]。

充电机器人自动驾驶技术主要是解决以下问题：

（1）面向全地图的全局路径规划：面向非结构化场景并基于已构建的地图设计最优路径。

（2）基于动态环境实时避障的局部路径规划：依托多源传感器（LiDAR、毫米波雷达）构建局部代价地图[5]，实现动态障碍物（如移动行人、突发障碍）的实时轨迹预测与避让，确保导航的安全性与连续性[6]。

（3）对于路径规划全过程的仿真验证体系：基于 ROS 与 Gazebo 仿真平台，构建物理属性参数，量化评估路径平滑度（曲率连续指标）、规划耗时等关键性能参数。

（4）对于传感器以及信息的融合处理研究。

（5）机器人的制作工艺、产品创新与装配技术。

在实际应用中，由于充电机器人复杂的作业环境或其他特殊情况，移动机器人自动驾驶技术受到限制，例如当车辆较多时行驶过程中可能与其他车辆或障碍物发生碰撞、机器人携带的大体积电池包可能与其他物体发生刮蹭等、在较为杂乱的环境中机器人的建图和路径规划工作可能不具有实时性和安全性等。

点云配准是自动驾驶中的关键一环，目的在于如何将激光雷达在不同位置下

采集到的点云数据统一到同一坐标系内[7]。激光雷达作为移动机器人环境感知的核心传感器，点云配准的精度直接决定 SLAM（同步定位与建图）系统的可靠性[8]。当前配准的主流方法包括以下三部分：基于特征的配准：提取点云特征（如 FPFH、ISS）并通过 RANSAC 剔除误匹配，但对噪声比较敏感且依赖特征提取的质量；基于概率模型配准：如一致性点漂移（CPD）算法[9]，将点集运动建模为概率密度函数，适用于非刚性场景；基于深度学习驱动配准：基于 PointNetLK 等网络架构实现端到端配准，显著提升了动态环境下的鲁棒性。而由于激光雷达采集的点云通常数据量较大且数据密度不均，伴随着噪声与冗余数据（例如指示牌、行人等引起的漏点或噪声）等问题，进行点云配准时不具有快速性、实时性，会严重影响几何特征表达、配准的效率与精度，进而影响 SLAM 建图的效率与精度[10]。

路径规划是指为系统选择一系列状态使机器人从初始状态到达期望的最终状态的过程。在计算机科学和机器人学的许多领域都有应用，如人工智能、工业自动化和移动机器人系统，而目前存在的路径规划算法仍存在着很多挑战，例如面对复杂环境时，规划的路径不具备快速性、不能满足机器人实际需求等，需结合应用场景作进一步优化。

解决机器人在作业时存在的各种问题，提高复杂环境下点云的配准精度与效率，为 SLAM 地图的构建提供高质量参数；路径规划方面，确保路径既符合机器人快速性、实时性需求的同时提供一定的安全保障具有非常重要的意义。

1.2 国内外研究现状

1.2.1 国内外充电机器人研究现状

上个世纪 60 年代，世界便开启了移动机器人的研究序幕。美国核管理委员会（NRC）曾预测，无人系统将成为 21 世纪的核心作战武器[11]。DARPA 于 2004 年发起 Grand Challenge 赛事，加速了自主导航技术的突破，催生了机器人从实验室走向实际应用[12]。随着新能源技术和新能源充电设备不断更新迭代，充电机器人的研究在结构创新与技术融合等方面有了更多突破。例如 Okunevich Iaroslav 等设计了一种新型充电机器人，采用倒 Delta 机制，结合卷积神经网络驱动的高保真触觉感知系统，实现了精确的三维定位，优化了电动汽车的自动充电过程[13]。

2023 年 Hoss Design USA 与 Car Charging Bot 合作，旨在通过开发一种无线充电机器人来革新电动车辆的充电方式，这项技术消除了传统充电方式中需要物理连接的限制，使充电过程更加便捷[14]。文献[15]研发了一种基于卷积神经网络的人机交互方式，通过手势结合视频会议技术，使用户能够通过手势指令控制充电机器人进行充电操作，提升了电动汽车的充电便利性。Mark Moran 在文献[16]中提到一款单臂自动充电机器人，基于 3D 摄像头与智能控制算法自动启动充电口，并正确插入充电器端口。

国内在无线充电技术及电动车充电解决方案方面的研究正不断取得突破。Zhang 等在针对充电机器人充电枪插拔方面提出了一种新的力控制策略，在实验室环境和 Tesla 模型 3 电动汽车上的实验表明,该方法有效实现了充电枪的安全平稳插入，但实验仅在静态实验室环境下验证，未考虑真实场景中的其他因素影响[17]。Song 等针对机器人的充电方式设计了一种基于磁耦合谐振的无线充电方法，实现了高效能量传输，解决了电动汽车需要频繁充电插拔充电枪的过程，但在复杂环境中可能存在着安全隐患[18]。2022 年由方杰教授团队与安徽中科源起科技有限公司合作开发的新能源移动充电机器人，如图 1-1(a) 、(b)所示。机器人通过分析常用的路径规划算法，在执行作业时构建安全等级评价体系，确保了机器人的行驶安全保障。

(a) 机器人底盘

(b) 智能移动机器人

图 1-1 中科源起移动充电机器人

1.2.2 点云配准研究现状

无人车通过搭载传感器捕获周围环境信息，激光雷达采集的点云数据是无人车进行环境感知并做出配准反馈的重要数据来源[19]。点云配准作为无人车自动驾驶中的关键一环，目的在于如何将在不同位置下采集到的点云数据统一到同一坐标系内，对高精地图制作、高精定位以及促进自动驾驶技术的发展等方面具有重

大意义[20-23]。

点云配准算法按照配准阶段可以分为粗配准与精配准。粗配准与精配准是一个不断迭代求解点云变换矩阵的过程，直到达到设定的误差阈值或迭代次数上限为止[24]。

点云配准主要有基于全局搜索思想[25]与基于几何特征描述[26-29]两种方法。其中基于全局搜索思想的配准算法主要有 RANSAC 算法[30]，由 Fischler 于 1981 年提出，该算法可以从大量噪声点与外点中估计数学模型，输出结果为内点最多的最佳数学模型，实现相对简单、易于理解。Nicolas Mellado 等提出的 Super-4PCS 算法有效降低了原始 4PCS 算法的时间复杂度，显著提升了配准效率，然而该算法在全局范围内搜索共面四点集的计算成本仍然相对较高[31]，文献[32]利用 ISS-3DSC 特征与 RANSAC 算法进行粗配准，在保持高配准精度的同时，具有较快的配准速度。但在某些复杂点云中，特征点的提取可能不够准确或稳定，会影响后续配准的精度和鲁棒性。

2003 年 Bibe 等[33]提出了 NDT 算法，主要用于同时定位和地图生成(SLAM)算法中二维平面点云数据的配准。该算法具有较高的计算效率且实用性较广，后续很多学者对其进行了改进。2006 年，NDT 算法打破了维度限制，被 Magnus-son 等[34]使用在三维空间中并提出了名为 3D-NDT 的算法，通过融合点云 RGB 影像实现点云自动精确拼接，较传统拼接方法有很大优势。

较为经典的精配准方法 ICP 算法由 Besl Paul J 于 1992 年提出[35]，该算法通过多次迭代求解对应点集与变换矩阵，对点云进行刚体变换，是基于最小二乘法的最优配准。但其配准精度根据点云质量的高低而变化，且需要提供良好的初值，否则会有陷入局部最优的风险。W. Guan 等[36]提出了一种改进后的 ICP 算法，使用双向 KD-tree 加速对应点对的搜索，大大降低了算法的运行时间，相比传统的 ICP 算法明显提升了算法运行效率。文献[37]使用 NDT-ICP 算法把两种算法结合在一起，兼顾了 NDT 与 ICP 的优缺点形成了互补，最终实验配准精度有所提高，成功完成了地图的构建，但是该算法在面对大量点云数据或点云质量不高时配准耗时较长、效率较低。张赵良等[38]基于 ISS 算法提取特征点，并使用 RANSAC 算法进行粗配准，最后结合 Nanoflann 算法加速的点到平面的 ICP 算法进行精配准，在配准速度与精度方面都有了提升，但在处理点云对应关系时没有考虑冗余

数据的影响。Chen 等针对点云配准需要较大重叠区域的问题，提出了一种使用高级结构信息的配准方法[39]，为数据采集提供了更大自由度，但该算法并不适用于复杂结构或不规则形状的点云。

概率配准方法通过松弛匹配机制构建点云间的概率关联模型，其核心在于支持一对多映射关系，扩展了传统刚性匹配的约束条件，增强复杂场景下的配准鲁棒性[40]。密度函数建模将点云空间分布转化为混合概率密度函数（如高斯混合模型），实现点云的非刚性对齐；引入松弛因子权衡内点与离群点的影响权重，降低噪声与局部缺失对配准精度的干扰。例如文献[41]提出的基于内核相关（Kernel Correlation，KC）相似性度量的配准模型，将点云对齐转化为最大化核空间相关性优化问题，适用于部分重叠点云的快速匹配。Myronenko 等基于一致的点漂移（Coherent Point Drift，CPD）算法，通过概率密度估计实现非刚性形变配准[42]，显著提升医学图像与动态物体的配准精度。Heng 等设计了鲁棒配准范式[43]，结合截断最小二乘损失函数与半定规划松弛技术，在离群点占比较高时仍能保持高配准精度，为自动驾驶等强干扰场景提供解决方案。

1.2.3 路径规划研究现状

新能源的发展正驱动着全球能源领域和交通运输业的根本变革，随着技术进步和政策支持，新能源汽车和无人充电车将在未来的交通体系中扮演越来越重要的角色。而路径规划作为无人驾驶车辆[44]的重要研究课题，主要有 Dijkstra 算法[45]、A*算法[46]、D*算法[47]、Flyod[48]、RRT 算法[49]、蚁群算法[50]等。

A*算法作为一种高效的路径规划算法，自从 1968 年由 Hart, Nilsson 和 Raphael 提出以来，一直在计算机科学的各个领域内得到广泛应用。A*算法可以视为是 Dijkstra 算法的一种优化扩展，它保留了 Dijkstra 算法的保证最短路径的特性，同时引入了启发式的优化，提升了算法效率[51]。

文献[52]针对野外无路网条件单一探索方向较慢的问题进行研究，提出使用双向 A*算法完成最优路径搜索，大大提高了工作效率。文献[53]对算法进行改进，在 8 方向搜索基础上，采用跳点搜索策略，虽然减少较多的扩展节点，仍存在大量的冗余拐点。文献[54]通过加入安全距离代价值，使规划路径与障碍物保持安全距离。文献[55]通过把全局路径规划算法 A*算法与局部路径规划算法 DWA 算法相结合，在遇到动态障碍物时能够及时进行回避，增强了机器人的灵活性。

路径规划作为自主导航系统的核心模块，为移动机器人或无人机提供从起点到终点的最优或次优路径[56]。需要解决的主要问题有以下四点：

（1）环境感知与建模：该模块决定了机器人能否准确识别周围环境、感知并构建环境地图，使机器人能够在复杂环境中有效行动。

（2）路径优化：如何找到最短或最安全的路径，避免障碍物并考虑动态变化，路径的质量高低将直接影响机器人的工作效率。

（3）实时计算与响应：在实时系统中快速计算路径、识别并响应突发事件（例如动态障碍物或意外情况），避免事故和损坏，是提高机器人的安全性的重中之重[57]。

（5）多目标与多约束问题：解决多目标与多约束问题使机器人在更复杂和动态的环境中做出合理决策，确保路径规划能够综合考虑多方面的因素，提升任务执行的效率、安全性和可靠性。

传统全局路径规划算法如 Dijkstra 算法和 A*算法通过构建栅格地图或拓扑图进行路径搜索，稳定性较强[58]。Dijkstra 算法遍历所有节点寻找最短路径，适用于静态环境，在大规模复杂场景中易陷入“维数灾难”。A*算法引入启发函数有效加快了搜索效率，但路径平滑性不足。近年来，智能优化算法（如遗传算法、蚁群算法、粒子群算法）通过模拟生物进化或群体行为实现全局搜索，显著提升了复杂环境下的规划能力。改进粒子群算法（IPSO）通过动态调整惯性权重和多群体协作，在无人机路径规划中表现出更高的收敛速度和避障成功率。

Dijkstra 算法基于图论模型，采用广度优先搜索策略，通过全局遍历计算起始节点到地图中所有节点的最短路径长度，其发散式搜索特性导致时间复杂度高达$O\left(n^2\right)$，在大型地图中容易面临计算资源瓶颈[59]。

A*算法作为启发式搜索的典型代表[60]，不仅记录当前节点至起点的实际代价$g\left(n\right)$，同时引入启发函数$h\left(n\right)$预估当前节点到目标节点的期望代价（如曼哈顿距离、欧氏距离），通过优化函数$f\left(n\right)=g\left(n\right)+h\left(n\right)$动态选择搜索方向，启发函数的设计直接影响算法最优性与计算效率，若$h\left(n\right)$始终小于真实代价，则可保证路径最优性；反之可能陷入局部最优解。

此外，混合路径规划方法（如栅格-模拟退火法、遗传-图搜索法）通过融合

多算法优势，兼顾全局最优性与实时性，是解决动态障碍物场景的重要方向。如今现有算法仍面临着实时性不足、多目标优化冲突（如路径长度与能耗平衡）、不能满足实际需求等挑战，需结合应用场景作进一步优化。

1.3 本文的主要研究内容

通过对智能移动机器人和充电机器人的发展进行综合分析，针对机器人在自动驾驶方面遇到的各种问题，包括在进行 SLAM 建图时激光雷达的常规点云配准算法配准效率低、质量差和容易受到噪声影响；在进行全局路径规划时，冗余节点较多、路径质量较低、机器人在拐点处的碰撞风险较高等，对点云配准算法和路径规划算法进行优化，借助相关开源软件和开源产品进行实际测试。本文研究内容主要包括：

（1）搭建移动机器人点云数据采集的仿真环境模型和物理模型：使用 SOLIDWORKS 建模软件构建仿真机器人模型，并输出为 URDF 文件到仿真环境中，添加激光雷达插件为后续点云采集部分奠定基础；基于 ROS 系统使用 GAZEBO 构建仿真环境模型，利用速度控制指令控制机器人完成数据采集。

（2）激光雷达点云配准算法研究：针对常规点云配准算法在处理复杂情况时遇到的各种情况（例如行人、指示牌等引起的噪声较多使得配准效果较差）进行优化，减少噪声点对算法的影响，提高配准效率和配准质量。

（3）智能机器人路径规划算法研究：全局路径规划旨在筹划一条由初始点通往目标点的大致路径，综合比较 Dijkstra 算法与常规 A*算法的优缺点，对 A*算法进行优化并设计地图进行仿真，根据实际需求对路径进行平滑优化。

（4）机器人实际测试：基于 ROS 环境及其强大的功能包与机器人底盘建立通讯，使用上位机控制机器人搭载思岚 A2 激光雷达对实验场地进行数据采集并建图。建立并订阅相关节点，利用 RVIZ 和 2D Nav Goal 功能调用改进后的规划器完成相关路径规划功能。

论文的主要结构安排如下：

第一章分析机器人以及充电机器人的主要发展历程，介绍本文的研究背景与意义；综合阐述了机器人点云配准和路径规划算法的国内外研究现状，提出了常规算法和部分已有算法存在的不足，并针对不足做出改进，最后概括本文的主要

研究内容和结构安排；

第二章有关机器人的仿真平台和仿真环境的搭建。本文机器人仿真测试均基于 Ubuntu20.04 系统的 ROS 环境，利用 GAZEBO 和内部自带模型设计仿真测试环境，使用建模软件创建 URDF 模型，搭建机器人的内部结构、设置动力学属性，再导入到 ROS 环境中为后续在 GAZEBO 和 RVIZ 中进行数据采集和导航测试奠定基础。

第三章有关激光雷达点云配准算法的研究。提出改进后的 ICP 点云配准算法，在粗配准阶段首先对点云数据提取 ISS 关键点并进行特征描述，使用 RANSAC 进行粗配准，获取大致精确的初始值。在搜寻两片点云的对应点对时，首先离散点进行判定，减少所需搜寻对应点对的数量，同时使用 KDtree 结构加快精配准运行效率，最后使用采集到的点云和公共数据集进行算法验证。

第四章对 Dijkstra 和 A*算法进行综合对比分析，对常规的 A*算法进行了改进，旨在设计出一条既能到达终点又符合机器人作业流畅性和安全保障的最优平滑路径。通过对搜索策略和评价函数进行优化，使估计代价更接近真实值，使规划路径更适应实际应用，最后进行算法的仿真测试与实验结果的分析。

第五章借助实际机器人底盘进行测试，基于 ROS 设计了完善的智能导航系统，使用不同的功能包对实验场地进行构图和导航测试，核心架构包含传感器、SLAM 建图层和路径规划层，将算法编写为 Python 节点导入全局规划器中，编写 Launch 文件启动调用节点进行测试。

第六章总结分析本文的研究成果和发现，并指出不足之处，提出可深入的研究方向和可改进的算法研究方案。

第 2 章 基于 ROS 的仿真平台设计与搭建

2.1 研究平台与方案设计

本文有关机器人仿真实验与研究均基于 Ubuntu20.04 系统下的 ROS melodic 环境，该环境具有以下优点：

（1）在开发环境方面，ROS Melodic 集成了 Gazebo 9、RVIZ 及 ros_control，能够支持仿真模型与实体机器人之间的无缝切换，使得图像处理的效率更高，便于实验的操作便捷性。并且包含了丰富的场景库，包括动态障碍场景和真实的实验室数字模型，能构建出更逼真的仿真场景。

（2）在算法库方面，该环境集成了成熟的导航框架、SLAM 功能包及运动规划算法，能快速搭建机器人的核心功能；还支持 Python 3 与 C++的混合编程，使用 Catkin 编译能实现算法的快速封装，为后续实际测试提供了有效的技术支撑。

（3）在通信方面，通过 Nomachine 实现上位机与机器人的可视化远程交互，支持跨平台（Windows/macOS/Linux）访问 ROS Melodic 开发环境，且延迟较低，能实时对机器人进行调试。

基于 ROS 环境主要对机器人模型、数据采集仿真环境进行搭建以及在实际实验场地对机器人进行建图和导航测试。通过速度控制指令控制小车采集不同位置下的点云数据，为后续对点云配准算法的研究做准备；基于 ROS 编写 Launch 文件调用手柄控制节点，遥控机器人底盘进行建图；调用路径规划功能包进行自动导航测试。

2.2 仿真小车模型搭建与参数配置

2.2.1 机器人 URDF 的模型搭建

URDF（统一机器人描述格式，Unified Robot Description Format）是用于描述机器人模型的一种 XML 格式，用于定义机器人在物理和视觉上如何存在，包括其关节、链接、传感器和其他组件的几何形状、质量和材料属性等。在 ROS 环

境中可识别出 URDF 格式中的机器人主要物理模型，并通过特定的标签属性定义机器人的各种信息，被广泛应用于模拟、运动规划和控制等环节。具体标签与定义方法如下：

<link>：机器人模型中每个 <link> 表示一个特定的部分，例如手臂、腿部或头部等。连杆的各个属性共同构成机器人的整体结构和功能，影响机器人在物理仿真中的表现，使得机器人能够执行各种任务。其中相关 Link 子标签以及相关功能如表 2-1 所示：

表 2-1 Link 子标签汇总

Link子标签	功能
visual	描述机器人 link 部分的外观参数
inertial	描述 link 的惯性参数
collision	描述 link 的碰撞属性

<joint>：用于定义机器人模型中两个<link>之间的连接关系。每个<joint>用来描述关节的属性，包括它的类型、运动方式、父链接和子链接等。其中 URDF 主要支持以下几种关节类型：

表 2-2 Joint 关节类型汇总

关节类型	描述
revolute	旋转关节，允许绕单一轴旋转
prismatic	平移关节，允许沿单一轴线性移动
fixed	固定关节，连接两个部件且不可移动
continuous	连续旋转关节，无角度限制（常用于旋转场景）
screw	螺旋关节，同时实现平移和旋转的复合运动

<joint>子标签用于详细描述关节的特性和行为，包括运动学、动力学等相关参数，对应的各个子标签以及功能如表 2-3 所示：

表 2-3 Joint 子标签汇总

Joint子标签	作用
<calibration>	指定关节的标准参考位置，以便进行精确校准
<dynamics>	描述关节的物理特性，包括阻尼和摩擦，影响运动表现

Joint子标签	作用
<limit>	确定关节运动的边界条件，包括角度、速度和力矩限制
<mimic>	明确关节与其他关节的联动性，使其互相影响
<safety_controller>	提供安全控制参数，防止过载和潜在危险

根据实际产品机器人尺寸需求，结合实验室实际结构、激光雷达等因素，对机器人 URDF 模型进行搭建，大致确定四轮机器人的相关尺寸如表 2-4 所示：

表 2-4 机器人大致属性

机器人属性	数值/cm
车长	90-95
车宽	70-75
车高	40-45

2.2.2 机器人 URDF 的物理参数配置

基于 ROS 环境可以直接解析 URDF 文件的 XML 格式，定义各个标签、参数描述机器人的仿真模型。按照上文描述的标签定义搭建 URDF 模型：

使用 SOLIDWORKS2023 软件对仿真机器人模型进行搭建，创建 6 个 tf 坐标分别为 base_link、lf_wheel_link、lb_wheel_link、rf_wheel_link、rb_wheel_link、lidar_link 分别作为小车的车身、左前轮、左后轮、右前轮、右后轮；其中 lidar_link、lidar_joint 作为激光雷达的载体。将其导出为 URDF 文件到 Ubuntu 系统中，并通过 xacro 文件为其添加激光雷达插件。

在终端中使用命令 catkin_ws 来创建工作空间 catkin_make，ROS 节点通常会通过参数服务器将 URDF 模型存储为 robot_description，并且可以根据需要作进一步扩展，添加更多细节或特性，比如摄像头、各种传感器等；通过 launch 文件，可以方便地启动相关的节点进行仿真和测试。小车编译环境搭建的流程图如图 2-1 所示：

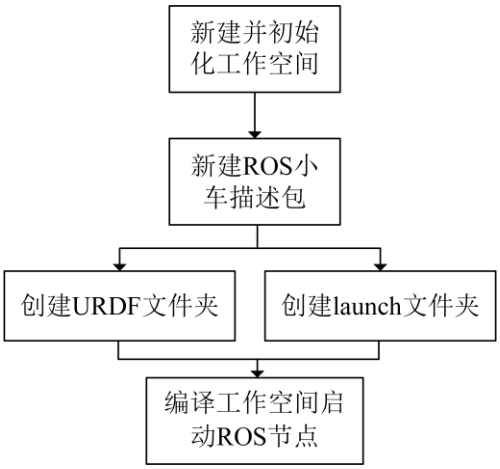


图 2-1 环境搭建流程图

在仿真实验中，需要同时启动多个 ROS 节点（如传感器节点、控制节点、规划节点、仿真环境等），此时使用 Launch 文件与 roslaunch 命令同时启动多个节点，并使用 <include> 标签将不同的子 Launch 文件组合在一起简化操作。在进行 launch 文件编写时需要遵守相关定义规则，相关标签与功能如表 2-5 所示：

表 2-5 Launch 子标签汇总

标签名称	作用
<launch>	Launch 文件的根标签，定义一个新的启动文件
<node>	启动一个 ROS 节点，指定节点所在的包和可执行文件
<param>	设置节点参数或全局参数，在运行时传递数据给节点
<include>	引入其他的 Launch 文件，可以实现模块化设计和复用
<group>	将多个节点或参数组织在一起共同设置环境和参数
<test>	运行一个测试节点，通常用于自动化测试或集成测试时

搭建好的机器人模型通过 ROS 环境下的 RVIZ 进行可视化，在 Launch 文件中添加 RVIZ 的相关标签与启动参数即可调用可视化界面。在 RVIZ 界面中不能直接看到加载的模型，需要为机器人添加模型显示选项 RobotModel，并修改 Fixed Frame 参数即可看到加载后的机器人 RVIZ 模型。调用 RVIZ 界面显示小车模型流程如图 2-2 所示：

图 2-2 Launch 文件启动流程

RVIZ 界面中的仿真机器人模型如图 2-3 所示：

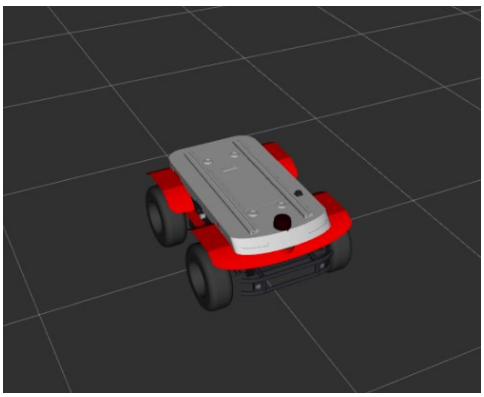


图 2-3 RVIZ 机器人可视化

小车的整体形状为长方体，转向结构采用阿克曼底盘的转向结构，激光雷达位于小车前部靠近下方的位置，形状设计为长方体。在 ROS 终端运行命令 rosrun tf view_frames 可以查看小车内部的 joint、link 等连接是否完整。小车主体与其他配件之间的连接图如图 2-4 所示：

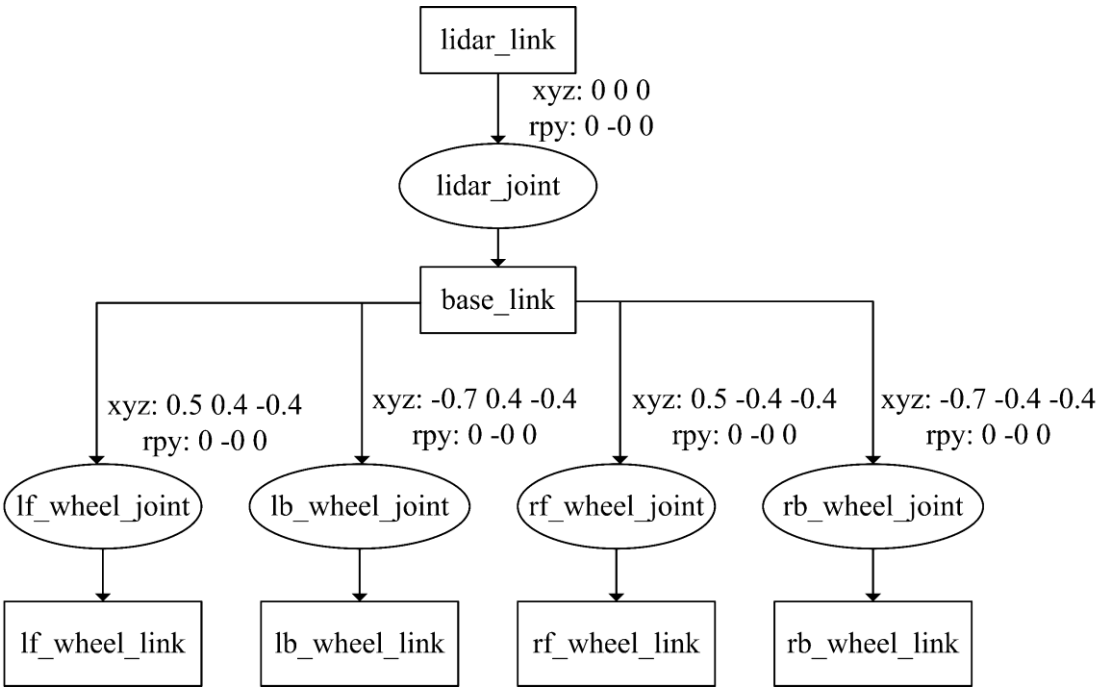


图 2-4 URDF 内部结构图

激光雷达等插件将添加在 URDF 文件模型中，有关激光雷达和摄像头的配置可以直接在 Github 中下载库文件，再根据具体需求修改雷达或者摄像头的相关参数即可，最后使用 catkin_make 命令编译工作空间确保相关插件已正确安装。

2.3 基于 ROS 环境的 GAZEBO 仿真环境搭建

GAZEBO 作为 ROS 系统中使用的默认机器人仿真软件，通过“gazebo”节点操纵仿真环境的属性，在环境中生成模型状态，使用 ROS 话题和服务发布和订阅仿真实验所需要的传感器数据、控制命令等。GAZEBO 提供多个仿真模型，可以模拟出真实复杂的室内外环境进行实验。除此之外，GAZEBO 还具有以下特点：

（1）与 ROS 系统的高度融合性、可扩展性；

（2）包含多种传感器模型供使用，包括激光雷达、摄像头、IMU（惯性测量单元）、GPS 等；

（3）具有真实的物理引擎支撑，更加便于相关机器人的仿真实验；

（4）是一个完全开源的软件，并且拥有活跃讨论社区解答用户各种疑问；

（5）GAZEBO 支持多种渲染引擎和图形界面，仿真场景更加 3D、立体。

安装并使用 GAZEBO 的步骤如下：

在 Ubuntu20.04 系统的 ROS 终端内安装 GAZEBO 软件库；在工作空间当中的 CMakeLists.txt 文件中添加依赖项；添加 GAZEBO 的相关功能代码；编译工作空间即可使用 GAZEBO。

本文主要使用 GAZEBO 软件对仿真实验环境进行构建，并模拟机器人在工作时应该具有的碰撞属性，使用的相关功能包如表 2-6 所示：

表 2-6 GAZEBO 功能包汇总

功能包名称	功能描述
gazebo_ros	GAZEBO接口封装、服务端启动、生成URDF模型
gazebo_msgs	Msg 和 Srv 数据结构便于与ROS进行通信
gazebo_plugins	模拟多种通用传感器插件
gazebo_ros_api_plugin	接口封装
gazebo_ros_path_plugin	路径绘制
gazebo_ros_control	与ROS控制框架集成控制机器人
gazebo_dev	自定义开发插件等

在软件中下载仿真模型数据即可进行环境的搭建，如图 2-5 所示为搭建所需的部分模型菜单：

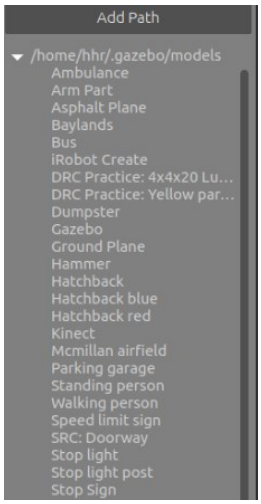


图 2-5 GAZEBO 模型菜单

一般情况下无人车的作业环境位于室外停车场或地下停车场内，为了符合实际应用场景及需求，本研究基于实际停车场与无人车搭建仿真环境模型，仿真停车场环境的搭建如图所示搭建的仿真环境如图 2-6 所示：

图 2-6 GAZEBO 仿真停车场

在此仿真环境中进行点云数据的采集，并验证点云配准算法的有效性。其中包含墙体、车辆、行人、指示牌等物体，环境中的模型具有真实物理碰撞属性，由于软件模型本身的细节因素，在进行数据采集时物体的点云信息一般呈现为规则形状。

2.4 本章小结

本章内容结合实际应用需求以及技术可行性等多方面因素，通过对各类仿真研究平台和硬件平台的研究与比较，挑选出适合本文研究目标与背景的仿真研究平台及相应的硬件实现平台，包括精度要求、实时性以及可扩展性等。

仿真模型部分采用统一的机器人描述格式 URDF 脚本文件进行编写，设计机器人的结构、关节、传感器配置以及动力学属性，为后续的仿真实验分析搭建平台；物理仿真环境的构建部分，通过 GAZEBO 的物理引擎、插件生态以及对 ROS 的集成，高度还原应用场景，为后续测试奠定基础。

第 3 章 基于改进 ICP 算法的点云配准研究

3.1 引言

同步定位与地图构建（SLAM）是机器人实现自主导航的核心框架。激光雷达通过发射激光脉冲并接收反射信号，生成离散的点云数据，SLAM 算法通过融合点云与运动估计，实时构建环境地图并跟踪机器人位姿。然而激光雷达输出的原始点云存在数据量大、密度分布不均且存在噪声等问题，对 SLAM 建图有较大影响。因此对点云配准技术的研究具有重要意义。

点云配准算法的精度与效率直接影响机器人定位与建图的可靠性。传统 ICP 算法因依赖初始位姿、易受噪声干扰且计算效率低，在复杂动态场景中易出现配准失败或误差累积问题。因此本章针对充电机器人作业场景特点，提出一种融合 ISS 特征提取、FPFH 描述子匹配、RANSAC 粗配准和改进 ICP 算法的粗精配准方法，结合 KDtree 加速搜索结构与设定最佳半径阈值，有效提升了点云的配准质量。

3.2 仿真模型与点云数据采集

基于第二章搭建的物理仿真环境和机器人仿真模型进行数据采集。通过给仿真机器人装配激光雷达插件，设置其扫描频率 10Hz，即每秒钟采集 10 帧点云数据，采集范围为 60 m，向周围环境发射激光束并按照预设的角度和频率进行扫描。接着向小车发布速度指令控制其进行移动，在不同位置采集不同帧数下的点云信息。

由于激光雷达采集到的距离信息不能直接作为点云信息使用，因此使用 laser_sender 功能包将/scan 话题采集到的信息转为 PointCloud 消息类型并保存在 PCD 文件中待进行下一步配准。为了使配准效果更加明显，在初始位置采集一帧点云，随后发布速度指令控制小车以 0.5 m 每秒的速度向前行进 5 m，再采集下一帧点云数据，分别采集到两片相隔 100 帧的点云数据作为目标点云和源点云，并为其添加一定的高斯噪声与随机均匀噪声。数据采集流程图如图 3-1 所示。待

配准的两帧点云数据如图 3-2 所示。

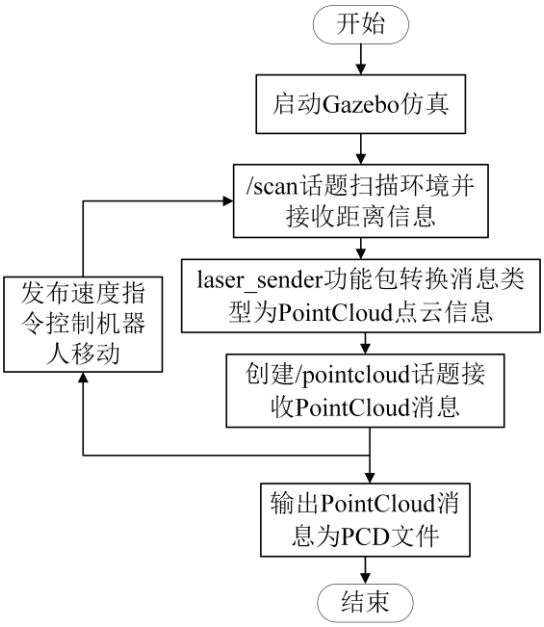


图 3-1 数据采集流程图

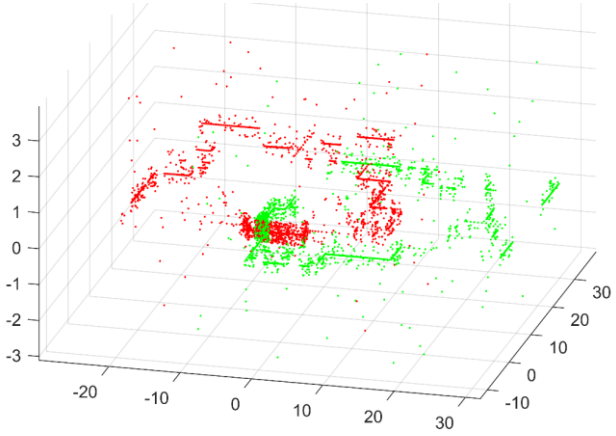


图 3-2 待配准的两帧点云数据

3.3 点云数据粗配准

3.3.1 ISS 关键点提取

ISS 算法首先设置一定的搜索半径 r 并计算每个点局部邻域的协方差矩阵，该矩阵反映了该点邻域内点云的形状和分布特性，通过比较代表邻域内点云在各个方向上的分布程度的协方差矩阵特征值来选取特征点，达到高质量的配准效果。

假设点云 $p_i=(x_i,y_i,z_i,)$ 中共含有 n 个点，$p_i=(x_i,y_i,z_i,)$ ；提取特征点的主

要流程如下：

（1）获取点云 P，为每个点 p_i 建立局部坐标系，并设置搜索半径 r。

（2）遍历 p_i 以 r 为半径的邻域内的所有点，计算其在邻域内的权重 ω_{ij}，权重表达式如公式（3-1）。

$$\omega_{ij}=\frac{1}{\left\|p_j-p_i\right\|},\quad \left\|p_j-p_i\right\|\le r \tag{3-1}$$

当邻域内的点距离 p_i 越近时具有更高的权重；领域内的点距离 p_i 越远时权重较小。

（3）计算每个点 p_i 的三维加权协方差矩阵 $\mathrm{cov}\left(p_i\right)$ 与特征值，并将其按照从大到小的顺序排列得到 $\left\{\lambda_i^1,\lambda_i^2,\lambda_i^3\right\}$；计算公式如（3-2）。

$$\mathrm{cov}\left(p_i\right)=\frac{\sum\limits_{|p_i-p_i|<r}\omega_{ij}(p_j-p_i)(p_j-p_i)^T}{\sum\limits_{|p_j-p_i|<r}\omega_{ij}} \tag{3-2}$$

设置一定的阈值 ε_1、ε_2，一般小于 1。所得的三个特征值主要反映了查询点的局部特征信息与相对重要性。当 $\lambda_i^1=\lambda_i^2>>\lambda_i^3$ 时，点云在其中一个维度上的信息量较少，局部呈现为平面；当 $\lambda_i^1>>\lambda_i^2=\lambda_i^3$ 时，点云在两个维度上的信息量较少，局部呈现为直线；若 $\left\{\lambda_i^1,\lambda_i^2,\lambda_i^3\right\}$ 相差无几，则该点周围没有相对明显的曲率变化，点云分布较为均匀，无显著特征。若点 p_i 同时满足公式（3-3）、（3-4）则被认为是关键点。

$$\frac{\lambda_i^2}{\lambda_i^1}\le\varepsilon_1 \tag{3-3}$$

$$\frac{\lambda_i^3}{\lambda_i^2}\le\varepsilon_2 \tag{3-4}$$

3.3.2 FPFH 特征描述子

FPFH 算法通过比较查询点与临近点之间邻域半径为 r 的局部空间差异，形成包含点特征信息的多维直方图。对查询点的 k 邻域内的几何属性进行描述，得

到每个点的简化点特征直方图（Simplified Point Feature Histogram，SPFH），再将邻域内所得 SPFH 进行加权统计得到该点的快速点特征直方图。算法流程如下：

（1）对于源点云和目标点云中的每个点 p_i 确定局部坐标系并计算其表面法线 n_i，确定每个点的邻域搜索半径并搜寻临近点 p_j。以 p_i 为原点计算各点的局部坐标值 (U,V,W)。计算公式如（3-5）：

$$\begin{cases} U = n_i \\ V = \dfrac{(p_i - p_j)}{\left\|(p_i - p_j)\right\|^2} \times U \\ W = U \times V \end{cases} \tag{3-5}$$

（2）使用欧拉公式计算邻域内各点的特征信息 (α,ϕ,∂)。计算公式如（3-6）：

$$\begin{cases} \alpha = v \times t \\ \phi = U \times \dfrac{p_i - p_j}{d} \\ \partial = \arctan(W \times n_i, U \times n_i) \end{cases} \tag{3-6}$$

$$d = \sqrt{\left\|p_j - p_i\right\|^2} \tag{3-7}$$

利用描述子的特征信息 (α,ϕ,∂) 与 d 绘制直方图并进行归一化处理。

（3）所得的 SPFH 只包含了特征点与其邻域点之间的特征关系，使用公式 3-8 对各个 SPFH 进行加权统计，得到最后包含邻域点与邻域点之间关系的 FPFH 特征描述子，提高了算法的描述准确性。$F(p_i)$ 计算公式如（3-8）：

$$F(p_i) = S(p_i) + \frac{1}{\omega}\sum_{j=1}^{k}\frac{1}{\omega_j}S(p_j) \tag{3-8}$$

$F(p_i)$ 为点 p_i 的 FPFH 值，$S(p_i)$ 为 p_i 的 SPFH 值，$S(p_j)$ 为 p_i 临近点 p_j 的 SPFH 值，ω_j 为查询点 p_i 到其临近点 p_j 的欧几里得距离，ω 为权重，与 ω_j 成正比。

3.3.3 RANSAC 粗配准

使用 RANSAC 算法进行点云粗配准，本质在于不断地对源点云和目标点云对应点集进行随机采样、求解刚体变换模型，设置最大迭代次数与拟合误差，从

而找到最佳变化模型。其主要步骤如下：

（1）随机选取源点云 P 与目标点云 Q 中的 3 个对应点对，假设 P 中选取的数据点为$\{p_1,p_2,p_3\}$，Q 中选取的点云数据为$\{q_1,q_2,q_3\}$，保证两片点云中 3 个点互不共线，并通过最小二乘法求解其变换矩阵。

（2）将 P 通过变换矩阵进行变换得到新的点云集 P'，计算公式如（3-9）：

$$P' = RP + t \tag{3-9}$$

式（3-9）中，R 为旋转矩阵，t 为平移向量。计算 P' 与目标点云 Q 之间的距离投影误差。在进行一次迭代后，将小于设定阈值误差的点记为内点，统计内点数目并记录变换模型。

（3）重复以上描述步骤至设定迭代次数，不断更新内点数量与变换模型，选择样本内点数量最多的模型为最佳数学模型。

3.4 点云数据 ICP 精配准

3.4.1 传统 ICP 算法

经过一系列粗配准后，源点云与目标点云之间仍然存在一定误差，基于粗配准提供的良好初值进行下一步精配准。传统 ICP 算法首先确立源点云与目标点云之间的对应点，本质在于求解对应点对之间的变换矩阵，求得使误差函数 $E(R,t)$ 最小的最优解 (R,t)。误差函数计算公式如（3-10）：

$$E(R,t) = \frac{1}{n}\sum_{i=1}^{n}\left\| q_i - (Rp_i + t) \right\|^2 \tag{3-10}$$

式（3-10）中，R、t 分别为旋转矩阵和平移向量，p_i、q_i 分别为点云源点云 P、目标点云 Q 中的点。算法主要流程如下：

（1）获取点云 P、Q，找出 Q 中与源点云 P 中每个点 p_i 的欧氏距离最近点作为初始对应点。

（2）使用奇异值分解（Singular Value Decomposition, SVD）方法计算最优变换矩阵，首先求得源点云 P 与目标点云 Q 的质心分别为 $\bar{p}$、$\bar{q}$，将两组数据进行归一化处理，计算公式如（3-11）、（3-12）：：

$$\begin{cases} \overline{p} = \dfrac{\sum_{i=1}^{n} \omega_i p_i}{\sum_{i=1}^{n} \omega_i} \\ \overline{q} = \dfrac{\sum_{i=1}^{n} \omega_i q_i}{\sum_{i=1}^{n} \omega_i} \end{cases} \tag{3-11}$$

$$\begin{cases} x_i = p_i - \overline{p} \\ y_i = q_i - \overline{q} \end{cases} \tag{3-12}$$

x_i 、 y_i 分别为归一化处理后的源点云与目标点云；假设变化矩阵 $G = x_i * y_i$ ，由 SVD 分解求得旋转矩阵 R、平移向量 t。计算公式如（3-13）、（3-14）：

$$R = VU^T \tag{3-13}$$

$$t = p - Rq \tag{3-14}$$

（3）根据所得的变换矩阵对点云 P 进行变换得到新的点集 $P' = Rp_i + t, p_i \in P$ 。重复以上步骤直到设定迭代次数上限 N_{max} 或者距离误差小于设定阈值时停止迭代。

3.4.2 改进 ICP 算法

ICP 算法的重点在于如何准确寻找源点云 P 与目标点云 Q 之间的对应点，激光雷达在扫描周围环境时，由于自身的采样噪声、环境因素干扰以及数据处理误差等，会产生大量的冗余数据或离散点，这些离散点在寻找对应点对时很难准确进行配对，从而降低了点云配准的质量，增加搜寻时间。为了提高配准质量与配准效果，在搜寻对应点对前首先筛选对实验有影响的离散点。判断源点云与目标点云离散点的标准与筛选步骤如下：

（1）获取源点云 $P = \{p_i, i = 1,2,\cdots,N_p\}$ 与目标点云 $Q = \{q_i, i = 1,2,\cdots,Nq\}$；

（2）构建 KDtree 进行半径搜索并设置每个点的搜索半径为 r，源点云中各个点 p_i 、 p_j 之间的距离为 $Dp_{ij}, (i \neq j)$ ；目标点云各个点之间的距离为 $Dq_{ij}, (i \neq j)$ 。使用欧氏距离表示如（3-15）、（3-16）所示：

$$Dp_{ij}=\sqrt{\left(x_{p_j}-x_{pi}\right)^2+\left(y_{p_j}-y_{pi}\right)^2+\left(z_{p_j}-z_{pi}\right)^2} \tag{3-15}$$

$$Dq_{ij}=\sqrt{\left(x_{q_j}-x_{qi}\right)^2+\left(y_{q_j}-y_{qi}\right)^2+\left(z_{q_j}-z_{qi}\right)^2} \tag{3-16}$$

（3）针对搜寻点 p_i，判断 p_i 半径为 r 的领域内是否存在其他点，遍历源点云中的所有点 $p_j,(i\neq j)$。若存在 Dp_{ij} 小于搜索半径 r，则点 p_i 半径为 r 的邻域内存在其他点，保留点 p_i 进行下一步配准；若不存在 Dp_{ij} 小于 r, 则认为 p_i 是离散点从源点云搜寻对应点集中剔除；目标点云的剔除方法同上，遍历目标点云中的所有点 $q_j,(i\neq j)$。若存在 Dq_{ij} 小于搜索半径 r，则保留点 p_i；若不存在 Dq_{ij} 小于 r, 则认为 p_i 为离散点从目标点云搜寻对应点集中剔除。

（4）输出处理后的源点云搜寻对应点集 $P^{'}=\left\{p_i,i=1,2,\cdots,N^{'}{}_p\right\}$ 与目标点云搜寻对应点集 $Q^{'}=\left\{q_i,i=1,2,\cdots,N^{'}q\right\}$ 进行 ICP 精配准。流程图如图（3-3）所示。

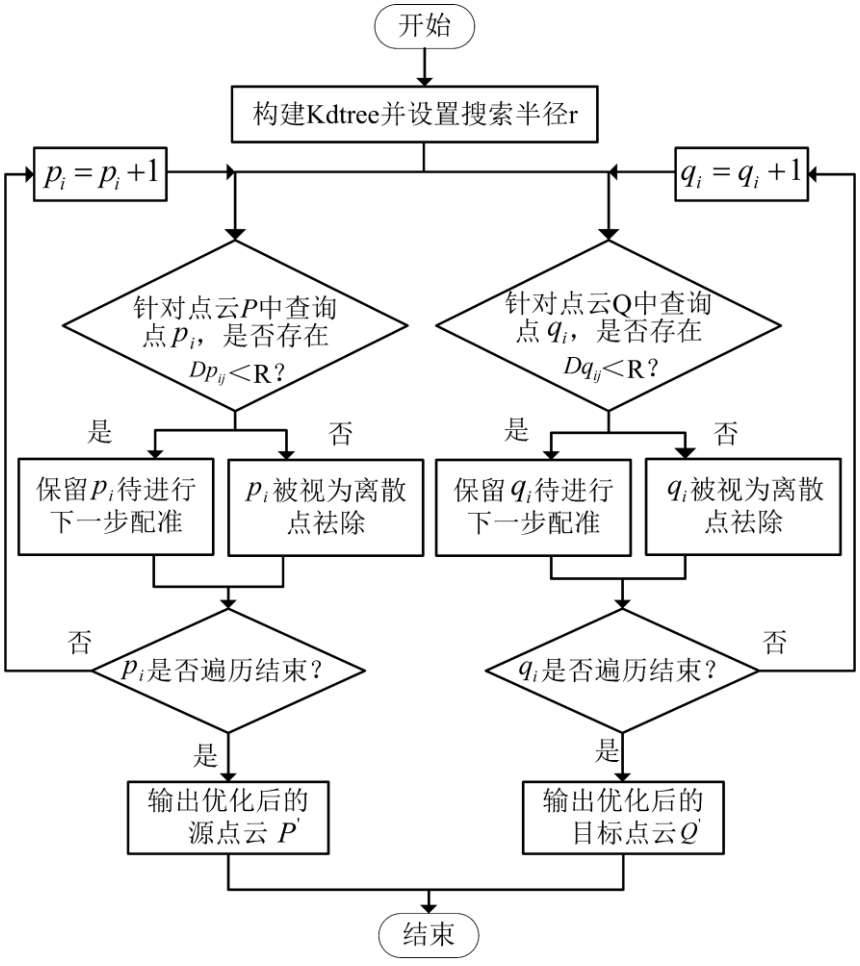


图 3-3 祛除离散点流程图

ICP 算法在搜寻对应点对时，需要遍历点云中的每个点，在面对点云复杂度较高、数据量较大的情况时，采用传统的线性查找方式会占用大量时间，降低算法效率，而 KDtree 结构的查询时间复杂度通常为$O(\log N)$，其中 N 为数据点的数量，随着点云数据量的增加，KDtree 的查询效率会发挥出更为明显的优势。

3.5 算法总流程

针对初始位姿要求高、配准效率低、受噪声干扰较大等问题，本文算法总流程如下。首先使用 ISS-FPFH 算法对源点云和目标点云进行特征点提取、计算点云特征描述子，再通过 RANSAC 算法剔除错误点对后对源点云和目标点云进行粗配准，为后续配准提供良好初值。在点云精配准寻找对应点对前，祛除离散点以减小对配准效果造成的影响，使用点到点的 ICP 算法进行精配准，通过 KDtree 加速寻找最近对应点对，最后实现源点云与目标点云的精确配准。算法的总流程图如图 3-4 所示。

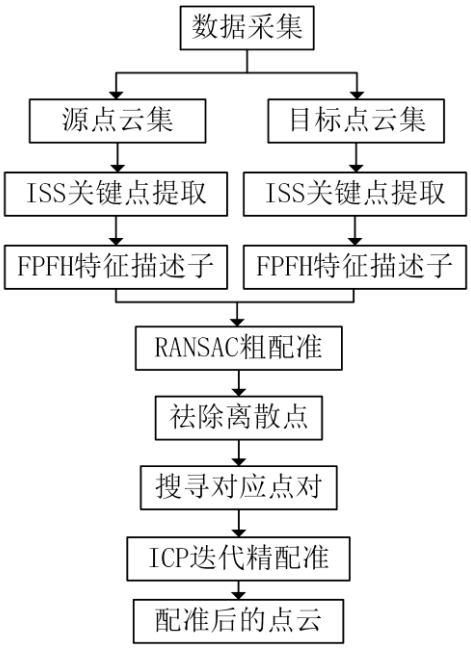


图 3-4 算法总流程图

3.6 仿真结果与分析

3.6.1 粗配准仿真实验分析

本实验采用基于第 2 章仿真环境采集到的相隔 100 帧的两片点云 P、Q，分别为源点云与目标点云。首先对其按照本文描述的方法进行点云粗配准，为后续配准提供良好初值。为了更好得比较实验结果，使用均方根误差（Root Mean Squared Error, RMSE）进行定量分析，其公式如（3-17）所示：

$$E_R = \sqrt{\frac{\sum_{i=1}^{n} p_i - q_i}{n}} \tag{3-17}$$

如表 3-1 所示，采集到的源点云和目标点云在添加一定的高斯噪声与随机均匀噪声后，使用 ISS 算法提取关键点大幅减少搜寻点的数量，加快粗配准的效率。提取到的关键点如图 3-5 所示。随后在提取到关键点的基础上对点云进行 FPFH 特征描述并匹配对应点，最后通过 RANSAC 算法进行粗配准得到变换后的源点云 P'。

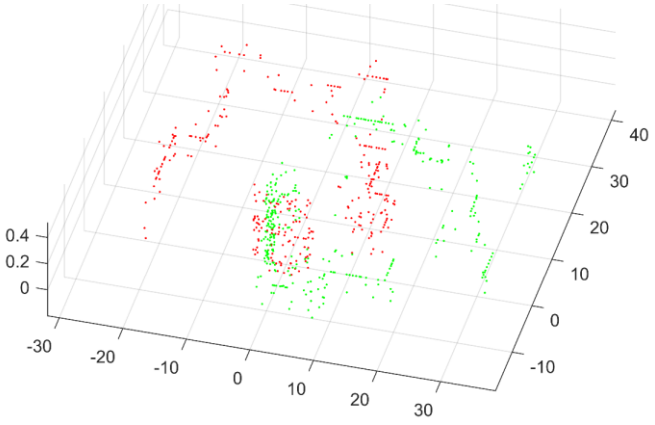


图 3-5 关键点提取

表 3-1 粗配准

	关键点数量	体素大小/m	配准时间/s	配准误差/m
源点云P	337	0.1	1.8146	0.9121
目标点云Q	309			

经过 ISS+FPFH+RANSAC 算法粗配准后的源点云和目标点云如图 3-6 所示。可见两片点云已实现了大部分的重叠，但仍然存在一定误差，在取得良好初值的

前提下对源点云和目标点云进行精配准，继续减小两片点云之间的配准误差以获得最佳配准效果。

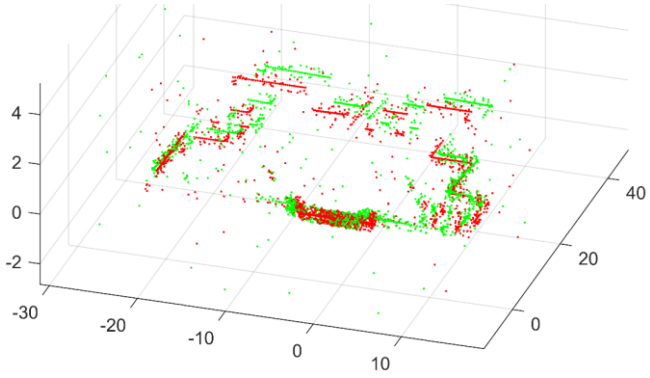
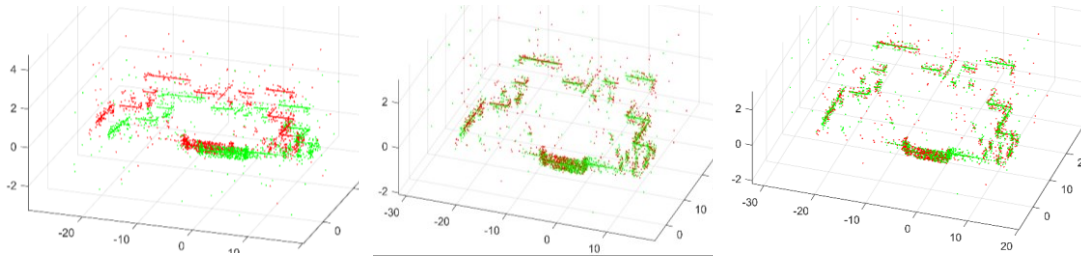


图 3-6 粗配准后

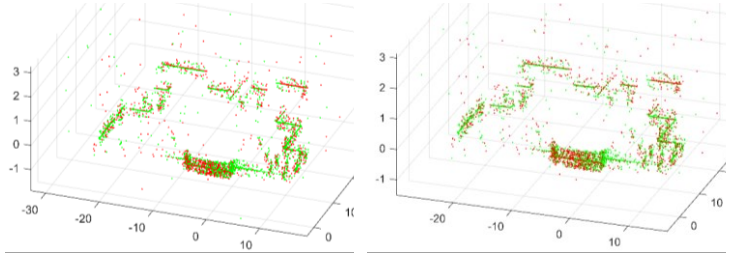
3.6.2 精配准仿真实验分析

获取粗配准初值后，使用改进后的 ICP 算法进行精配准，在搜寻源点云与目标点云对应点对前首先祛除离散点。算法的配准效率取决于保留下的点云质量，因此离散点搜寻半径阈值的选取将影响到算法的配准质量。如果半径过小，可能导致关键信息丢失，使得配准结果不准确；如果半径过大，则可能引入不必要的噪声，同样会影响配准精度。

设置离散点搜寻半径阈值 r 分别为 0.3 m、0.4 m、0.5 m、0.6 m、0.7 m 进行精配准实验。不同搜索半径下的精配准结果与比较分析分别如图 3-7 和表 3-2 所示。



(a) 0.3 m　　(b) 0.4 m　　(c) 0.5 m



(d) 0.6 m　　(e) 0.7 m

图 3-7 不同半径阈值精配准结果

表 3-2 各半径阈值配准结果对比

半径阈值/m	剩余对应点数量/个		配准误差/m	配准时间/s
	P	*Q*		
0.3	838	870	1.5551	0.0941
0.4	1460	1453	0.6486	0.1302
0.5	1907	1920	0.6308	0.1638
0.6	2120	2229	0.6319	0.1824
0.7	2260	2260	0.6410	0.2196

由表 3-2 和图 3-7 所示可知，当 r 为 0.3 m 时，源点云与目标点云对应点大幅度减少，使得点云关键信息丢失，配准误差较大；随着半径阈值不断增大，保留对应点个数随之增加，配准时间增加；当 r 为 0.7 m 时，阈值过大超出数据点之间的合理距离范围，导致配准算法无法满足要求；通过结果比较分析，选取 0.5 m 作为最佳半径阈值，保留点云关键信息的同时配准误差最小。图 3-8 所示为祛除离散点后待精配准的源点云与目标点云。

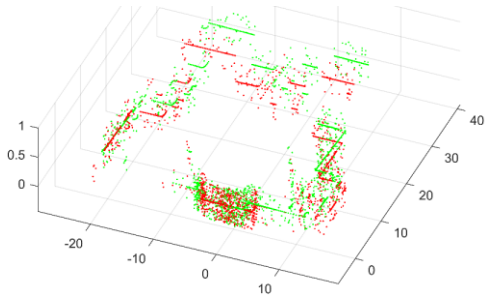
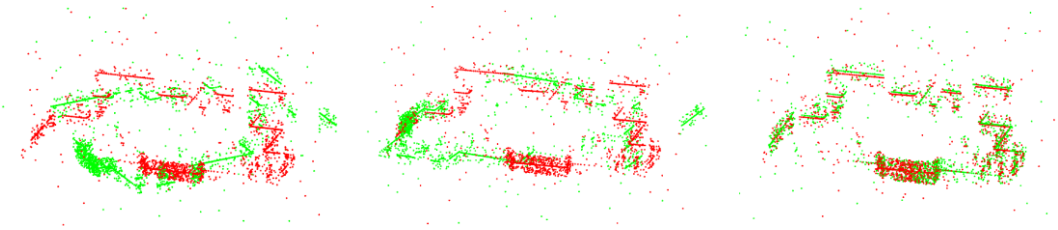
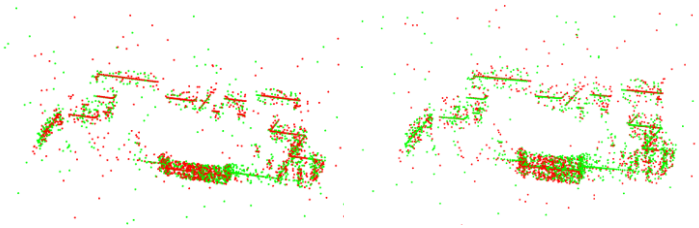


图 3-8 离散点祛除

为了验证改进后的 ICP 算法在采集到的点云数据模型上的有效性以及添加噪声对整体匹配的影响，分别使用传统 ICP 算法、文献[18]提出的 NDT-ICP 算法、ISS-FPFH-RANSAC-NDT（简称 IFR-NDT）算法、ISS-FPFH-RANSAC-ICP（简称 IFR-ICP）算法以及本研究配准算法（简称 IFR-DICP）的仿真实验结果进行比较分析。各个精配准算法结果分析比较如图 3-9 所示。除传统 ICP 算法外，其他算法虽然都在粗配准提供的基础上完成了精配准，但配准效果各不相同。



(a) 传统 ICP　　(b) NDT-ICP　　(c) IFR-NDT



(d) IFR-ICP　　(e) IFR-DICP

图 3-9 各精配准算法对比

图 3-9(a)所示为传统 ICP 算法对初始点云进行配准后的结果，可见由于算法对点云初始位姿的依赖性较高，使得实验结果陷入局部最优解，配准误差较大；图 3-9(b)尽管 NDT 算法对初值位姿的敏感度较低，但 ICP 算法对其粗配准的配准效果仍然有较高的依赖性，使得配准误差达不到预期的效果。图 3-9(c)、3-9(d)、3-9(e)分别为 NDT、ICP 以及改进后的 ICP 算法在经过本文粗配准算法后的精配准效果，可见三种算法在粗配准的基础上进行了进一步配准，输出源点云与目标点云基本重叠，配准误差较小。

如表 3-3 所示为各种算法对目标点云和源点云分别进行十次配准试验后取平均值的实验结果。在对应点搜寻数量方面，传统 ICP 算法、NDT-ICP 算法与 IFR-ICP 算法搜寻数量相同，本研究算法在筛选离散点时设置搜寻半径 r 为 0.5 m，使得源点云搜寻数量减少 353 个，目标点云搜寻数量减少 340 个，分别占其点云总数目的 15.62%、15.04%，有效减少了所需搜寻的对应点数。

精配准时间方面，本文算法较于传统 ICP 算法、NDT-ICP 算法、IFR-ICP 算法分别降低了 90.57%、86.86%、90.38%；较于 IFR-NDT 算法虽然精配准时间增加了 19.65%、但配准误差减少了 10.05%。

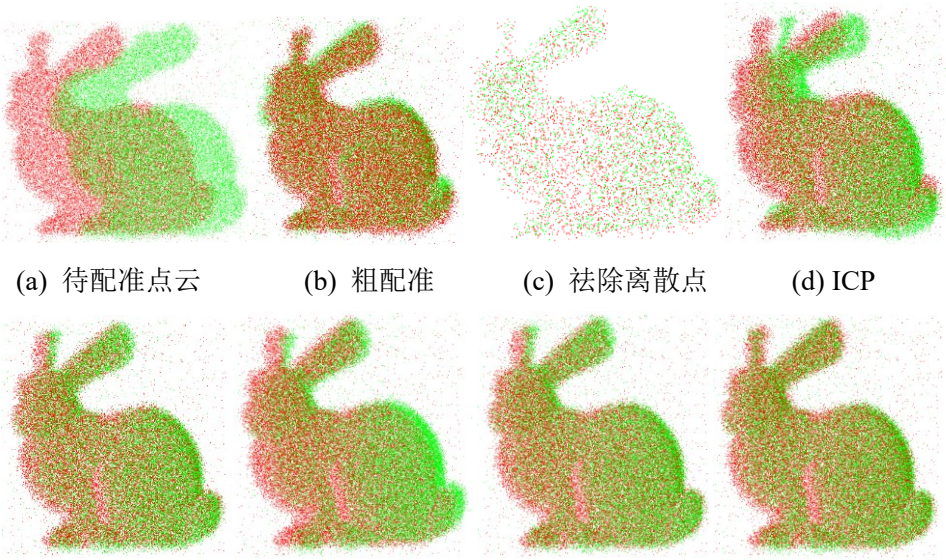
配准误差方面，由于传统的 ICP 算法对点云初始位置的要求偏高，且在面对大量数据时耗时较长、算法效率较低，而 NDT 算法虽然效率较高，但本身受到初始位姿的局限。因此 ICP 算法、NDT-ICP 算法虽然在一定程度上提高了点云配准精度，但误差仍然较大。而 IFR-NDT 算法、IFR-ICP 算法、本文算法与粗配准效果相比，配准误差分别降低了 0.2108 m、0.2711 m、0.2813 m；本文算法较前两种算法配准误差分别降低了 0.0705 m、0.0102 m，配准精度分别提升了 10.05%、1.59%。综上可得，改进后的点云配准算法配准时间较低、配准精度较高，总体质量上优于其他几种算法。

表 3-3 精配准算法分析

配准算法	搜寻对应点数量/个		迭代次数	配准误差/m	配准时间/s		
	P	Q			粗配准	精配准	总时间
ICP	2260	2260	18	2.1794	—	—	1.7374
NDT-ICP	2260	2260	38	1.9392	0.1164	1.2566	1.3730
IFR-NDT	—	—	19	0.7013	1.8146	0.1369	1.9515
IFR-ICP	2260	2260	26	0.6410	1.8146	1.7029	3.5175
IFR-DICP	1907	1920	8	0.6308	1.8146	0.1638	1.9784

3.7 公共数据集测试

使用斯坦福大学数据库中的 Bunny000、Bunny045 点云进行算法测试，如图 3-10 所示。



(a) 待配准点云 (b) 粗配准 (c) 祛除离散点 (d) ICP

(e) NDT-ICP (f) IFR-NDT (g) IFR-ICP (h) IFR-DICP

图 3-10 Bunny 配准对比

表 3-4 Bunny 算法对比分析

算法	E_R/cm	配准时间/s		
		粗配准	精配准	总时间
ICP	30.58*10-4	—	129.04	129.04
NDT-ICP	29.01*10-4	0.15	122.13	122.28
IFR-NDT	29.34*10-4	37.00	0.33	37.33
IFR-ICP	25.24*10-4	37.00	123.37	160.37
IFR-DICP	24.06*10-4	37.00	3.89	40.89

原数据集的点云个数分别为 40256、40097，为了验证算法在实际应用场景的

有效性，分别为其添加标准差为 0.002 的高斯噪声和 1000 个随机噪声，经过处理后的目标点云 Bunny000 和源点云 Bunny045 的点云数量分别为 41256、41097，如图 3-10(a)所示，其中红色点云为目标点云，绿色点云为源点云。

图 3-10(b)为经过本文粗配准算法后得到的待精配准的两片点云；图 3-10(c)为祛除离散点后的点云，同时保留下的源点云和目标点云的数量分别为 5623、5481，分别占原点云数量的 13.63%、13.34%，经过离散点祛除后的点云在保持原有特征的基础上极大减少了所需搜寻的对应点对数量；图 3-10(d)、图 3-10(e)、图 3-10(f)、图 3-10(g)、图 3-10(h)分别为基于各算法进行配准后的结果图。

其中 ICP 算法配准误差较大，配准质量较差；本文算法相对于其他算法配准误差最小，配准精度最高，两片点云基本实现重叠。由表 3-4 可知，本文算法较传统 ICP 算法配准时间上减少了 68%、配准误差上减少了 21.32%；较 IFR-NDT 算法配准时间上略有增加，但配准误差减少了 18%。

在面对复杂点云情况时，传统 ICP 算法与 NDT-ICP、IFR-ICP 算法配准时间较长且配准误差较大，IFR-NDT 算法虽然配准时间稍短，但受各种条件限制，使得配准误差较大；本文算法较于其他算法，在面对众多噪声干扰的情况下配准时间较低、配准误差最小，在实际应用场景中更能满足算法抗噪声干扰强、配准效率高的需求。

3.8 本章小结

本章基于搭建的仿真模型与环境采集到的点云数据进行点云配准实验，利用 ISS 算法提取点云特征点并进行 FPFH 特征描述，再结合 RANSAC 算法剔除错误点对并进行粗配准，为后续精配准提供良好初值，最后使用改进后的 ICP 算法完成精配准。实验结果表明，本文算法较于传统 ICP 算法、NDT-ICP、IFR-ICP 算法在精配准时间上分别降低了 90.57%、86.86%、90.38%；配准误差分别减小了 1.5486 m、1.3084 m、0.0102 m，配准精度分别提升了 71.06%、67.47%、1.60%；较于 IFR-NDT 算法总配准时间有所增加，但配准误差减少，配准精度提高了 10.05%。改进后的点云配准算法配准时间较低、配准精度较高，总体质量上优于其他几种算法，更加契合无人车在实际道路环境中的高精度定位与地图匹配需求。

第 4 章 基于改进 A*算法的路径规划研究

4.1 引言

针对机器人在进行路径规划时算法不具备安全保障、冗余节点较多、计算量较大等问题进行研究，在常规 A*算法的基础上进行改进。首先对 A*算法的启发式函数进行改进，确保路径大致搜索方向趋向于目标点；其次对正向搜索的节点进行判断，减少冗余节点的浪费，并对子节点的临节点进行调查，为小车提供行驶安全保障；最后通过 B 样条曲线对生成的路径平滑处理，增强机器人在拐弯处的流畅性。研究结果表明，改进后的 A*算法，目标点无论是在空旷处还是在复杂环境中，都能很好的规划出一条可行的安全路径。

4.2 仿真应用场景设计

智能无人车在进行路径规划前，首先需要对作业环境进行扫描，本研究采用仿真栅格式的地图模型，如图 4-1 所示。栅格地图具有适应能力强、应用范围广等优点，把车载传感器接收到的信息抽象化二维地图。以封闭区域 P 的左上方为坐标原点，水平方向和垂直方向分别为 X 轴和 Y 轴，将地图 P 划分为 30*40 的大小相等的正方形栅格图，每个栅格使用不同的数字代表当前状态，其中 1 代表通畅状态，小车可以自由通过，在地图上显示为白色方块；2 代表静态障碍物，小车不能通过，在地图上用黑色方块表示；停车位用青色方块表示，状态与静态障碍物相同，小车不能通过；绿色和红色分别代表起点和目标点。如图 4-1 所示:

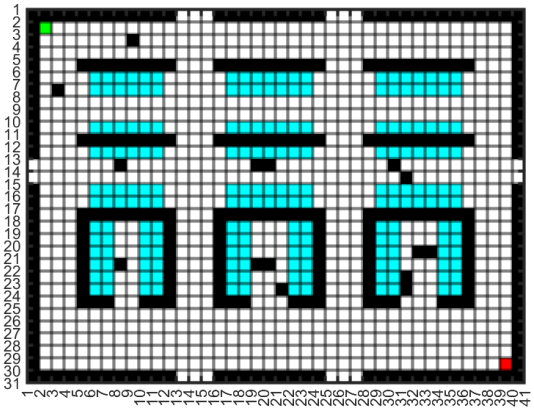


图 4-1 环境模型

每个栅格都具有线性索引和行列索引两种索引指标，如图 4-2 所示。

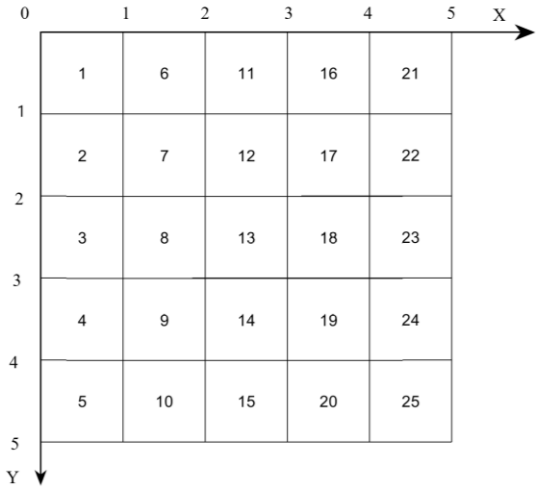


图 4-2 线性索引与行列索引的关系

在区域 P 中，水平方向的栅格个数 $n_x = X_{\max}/\alpha$；竖直方向的栅格个数 $n_y = Y_{\max}/\alpha$；其中 α 为小车的移动步长。$S = \{1,2,3,\ldots,n_x * n_y\}$ 为栅格的线性索引集，每个栅格的行列索引 (X_i, Y_i) 与指标 i 的关系计算公式如（4-1）、（4-2）：

$$X_i = ceil\left(\frac{i}{n_y}\right) \tag{4-1}$$

$$Y_i = (i-1) \bmod n_y + 1 \tag{4-2}$$

假设小车在封闭区域 P 中自由移动，其中有若干个静态障碍物和不可通过的停车位，小车的体积占据一个方格，考虑到在真实应用场景中小车的灵活性，采用欧几里得距离作为估计代价来计算总代价，小车可以向临近的 8 个方向进行移动，横向纵向每移动一步的代价设为 1。

4.3 路径规划算法研究与仿真

4.3.1 Dijkstra 算法

Dijkstra 算法是一种经典的最短路径算法，该算法以起始点为中心，逐层向外扩展，旨在每一步都寻求当前最优解，直到最终扩展到目标点。基本思想是通过维护一个优先队列，持续选择当前距离起始点最近的节点进行扩展，从而逐步找到从起始点到所有其他节点的最短路径。

算法的具体过程如下：

（1）设起点为 S 点，计算周围的 8 个节点，首并设置 S 点为父节点，规定上下左右节点的步长为 1，所需的代价值为 1；向 4 个对角线行走的步长为 2，所需的代价值为 2；将父节点周围 8 个节点分别标记：A、B、C、D、E、F、G；如图 4-3 和 4-4 所示：

	2	1	2	
	1	S	1	
	2	1	2	

图 4-3 邻近 8 节点代价值

A	B	C
H	S	D
G	F	E

图 4-4 邻近 8 节点指示图

创建数组 openList 和 closeList 分别存放待计算路径的节点已确定路径的节点；计算公式如（4-3）、（4-4）：

$$openList = \begin{cases} B_S = 1 \\ D_S = 1 \\ F_S = 1 \\ H_S = 1 \\ A_S = 2 \\ C_S = 2 \\ E_S = 2 \\ G_S = 2 \\ S_S = 0 \end{cases} \tag{4-3}$$

$$closeList = \{S = 0 \tag{4-4}$$

（2）从步骤（1）所得的 openList 中取出具有最小路程的点 B_S 作为当前父节点，并将该点从 openList 中移除，存放入 closeList 中，按照步骤（1）继续计算该点周围的 8 个节点。如果当前计算节点已经存在于 openList 中，代表该点已经计算过，选取代价值较小的方案替换原节点代价值（更新路径）；此时更新后的 openList 与 closeList 分别为公式（4-5）、（4-6）所示：

$$openList = \begin{cases} C_S = 2 \\ E_S = 2 \\ F_S = 1 \\ H_S = 1 \\ A_S = 2 \\ C_S = 2 \\ E_S = 2 \\ G_S = 2 \\ B_{BS} = 2 \\ A_{BS} = 3 \\ C_{BS} = 3 \end{cases} \tag{4-5}$$

$$closeList = \begin{cases} S = 0 \\ B_S = 1 \end{cases} \tag{4-6}$$

（3）接着将 F_S 节点作为下一中心节点计算邻近节点代价值，重复步骤（2）计算代价值并更新 openList 与 closeList；

（4）将 H_S 节点作为下一中心节点计算邻近节点代价值，重复步骤（2）计算代价值并更新 openList 与 closeList；

（5）将 D_S 节点作为下一中心节点计算邻近节点代价值，重复步骤（2）计算代价值并更新 openList 与 closeList；

（6）将 A_S 节点作为下一中心节点计算邻近节点代价值，重复步骤（2）计算代价值并更新 openList 与 closeList； 此时更新后的 openList 与 closeList 分别为（4-7）、（4-8）所示：

$$openlist = \begin{cases} C_S = 2 & B_{BS} = 2 & F_{FS} = 2 & H_{HS} = 2 & D_{DS} = 2 \\ E_S = 2 & A_{BS} = 3 & G_{FS} = 3 & A_{HS} = 3 & C_{DS} = 3 \quad A_{AS} = 4 \\ G_S = 2 & C_{BS} = 3 & E_{FS} = 3 & G_{HS} = 3 & E_{DS} = 3 \end{cases} \tag{4-7}$$

$$closeList = \begin{cases} S = 0 \\ B_S = 1 \\ D_S = 1 \\ F_S = 1 \\ H_S = 1 \end{cases} \tag{4-8}$$

（7）以此类推重复以上步骤，直到遍历所有待计算的节点。

以上 Dijkstra 算法步骤可以用以下图片表示：

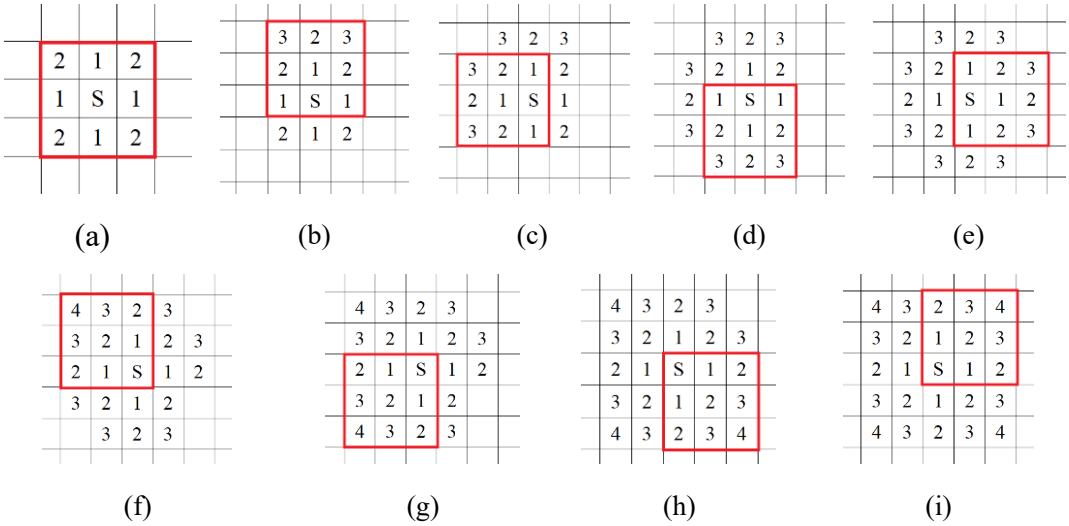


图 4-6 Dijkstra 算法步骤

综上所述，归纳出 Dijkstra 算法的流程图如图 4-7 所示：

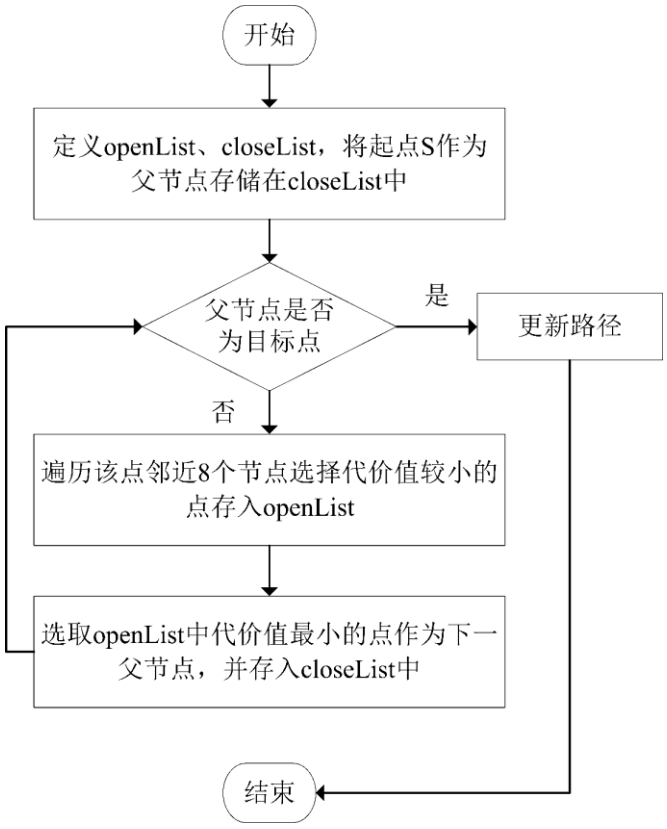
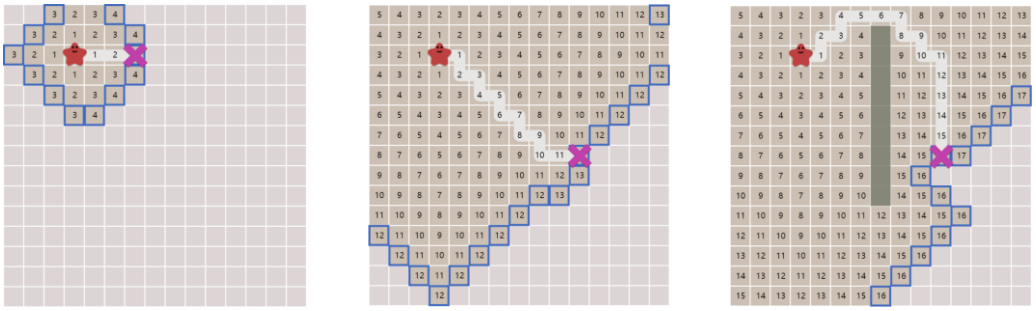


图 4-7 Dijkstra 算法流程图

使用网页“A Pathfinding Visualizer*”可以清晰看出 Dijkstra 算法的具体寻路步骤。如图 4-8 所示，五角星为起点、“X”标记为目标点。当起点与目标点相距较近时，Dijkstra 算法在起点与目标点距离较近的情况下能够迅速收敛到目标节点，并找出路径；而当起点与目标点之间的距离较远或者存在部分障碍物时，

算法需要遍历更多的节点以找到最短路径。这会增添更多的计算以及更长的执行时间，影响运行效率。



(a) 起点与目标点较近　　(b) 起点与目标点较远　　(c) 遇上障碍物时

图 4-8 不同情况下的 Dijkstra 算法

4.3.2 A*算法

A*算法的重点在于当前节点对起始点和目标点的代价值估计，较于 Dijkstra 算法通常能更准确更迅速规划最优路径。常规 A*算法的代价函数如公式（4-9）：

$$f(n)=g(n)+\omega(n)\cdot h(n) \tag{4-9}$$

启发式函数 $h(n)$ 代表从当前节点行驶到目标点的估计代价，$g(n)$ 为起始点到当前节点的代价称为实际代价，二者之和作为总代价组成 A*算法的代价函数，通过优化代价函数可以实现规划路径的搜索方向。由于实际代价 $g(n)$ 是已知固定的，而估计代价接近于实际值，所以很多学者选择对估计代价 $h(n)$ 进行优化，达到改进算法的目的。以下是常规 A*算法的具体步骤：

（1）假设同样规定上下左右节点的步长为 1，所需的代价值为 1；向 4 个对角线行走的步长为 2，所需的代价值为 2；将父节点周围 8 个节点分别标记：A、B、C、D、E、F、G；创建数组 openList 和 closeList 分别存放待计算路径的节点已确定路径的节点。

（2）将起点 S 作为父节点存入 closeList，根据公式计算起点 S 的邻近 8 个子节点（障碍物节点除外）的函数代价值 $f(n)$ 并存入 openList 中去；

（3）选取 openList 中代价值 $f(n)$ 最小的节点作为下一父节点存入 closeList 中，并计算邻近 8 节点的函数代价值（障碍物节点和已存在 closeList 中的节点除外），选择代价值较小的方案更新 openList；

（4）取 openList 中代价值 $f(n)$ 最小的节点作为下一父节点，重复以上步骤；

（5）如此循环遍历，直到 closeList 中存在目标点 G，输出最优路径与函数代价值。以上步骤可以表示为图 4-9：

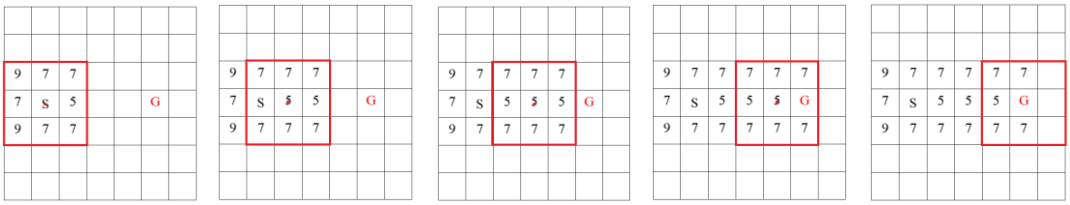


图 4-9 常规 A*算法步骤

常规 A*算法的流程图如图 4-10 所示：

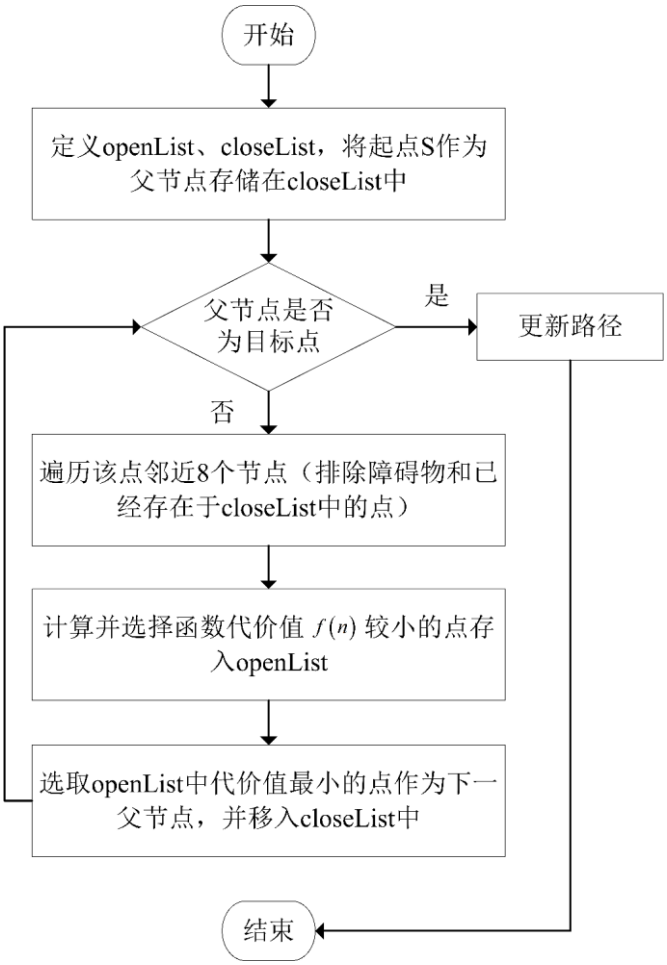


图 4-10 常规 A*算法流程图

针对搜索空间较大或复杂的情况，A*算法使用启发式函数能够有效减少冗余节点的数量，优化路径的搜索过程，通常比 Dijkstra 算法搜索效率更快。但在机器人导航或动态环境下的移动却存在一些不足。

A*算法主要依赖于启发式函数来估计从当前节点到目标节点的距离，当启发式函数不够精准或不符合实际情况，会导致找到的路径并非最优，影响算法效率。在处理复杂环境时，如狭窄通道或密集障碍物空间，A*算法没有充分考虑机器人的体积大小，无法有效避免碰撞。由于算法本身并未内置处理碰撞检测和避障的机制，机器人可能会误入障碍物覆盖的区域，因此需要在 A*算法内部追加障碍物检测算法，提高算法的实际应用能力。

4.4 基于 A*算法的优化

为了减少不必要节点的资源浪费，提高无人车行驶作业效率，确保机器人安全行驶，对常规 A*算法进行改进。首先对评价函数进行优化，使估计代价值大致接近于真实值；判断起点和终点的位置，确定既定的正方向，从而控制选取节点的方向，达到减少冗余节点数量的效果；同时路径搜索策略由单向搜索增加为双向搜索，减少路径规划过程所需时间；考虑到无人车在行驶过程时的碰撞风险，在进行对角移动时会对相邻的节点进行判断，避免由于体积碰撞问题带来的安全隐患，为小车提供一定的安全保障。

4.4.1 基于 A*算法评价函数优化

预估代价的评价函数通常使用曼哈顿距离与欧几里得距离作为评判标准，如图 4-10 和 4-11 所示分别为从点 A 到点 B 的曼哈顿距离与欧几里得距离示意图；曼哈顿距离与欧几里得距离表达式分别如公式（4-11）、（4-12）所示。

$$h(n)=abs(x_n-x_m)+abs(y_n-y_m) \tag{4-10}$$

$$h(n)=\sqrt{(x_n-x_m)^2+(y_n-y_m)^2} \tag{4-11}$$

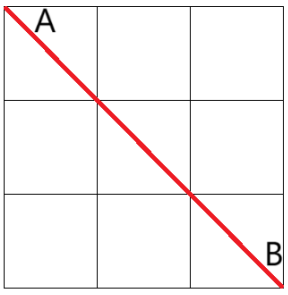


图 4-11 欧几里得距离

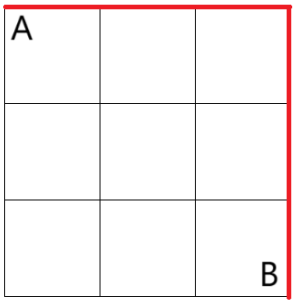


图 4-12 曼哈顿距离

无人车在面对复杂环境时，并不能始终按照径直行驶，因此当使用曼哈顿距离作为启发式函数计算标准时，得到的从当前节点 p 到目标点 G 的预估代价 $h(n)$ 会比实际代价偏大；而采用欧几里得距离得到的 $h(n)$ 要比实际代价值要小，本研究采用优化后的评价函数用来计算预估代价，介于曼哈顿距离与欧几里得距离二者之间，更能准确估算出当前节点行驶到终点的相对复杂的对角线距离，减小与实际值的误差。代价函数如公式（4-12）、（4-13）。

$$f(n)=g(n)+\omega(n)\cdot h(n) \tag{4-12}$$

$$\begin{cases} h_d=\min\left(abs\left(G_x-P_x\right),abs\left(G_y-P_y\right)\right) \\ h_s=abs\left(G_x-P_x\right)+abs\left(G_y-P_y\right) \\ h(n)=\sqrt{2}h_d+\left(h_s-2h_d\right) \end{cases} \tag{4-13}$$

其中 G_x 、 G_y 分别为终点的横纵坐标， P_x 、 P_y 为当前节点的横纵坐标。

在估计代价 $h(n)$ 的前面添加权重系数 $\omega(n)$ ，控制搜索节点的大致朝向，调整预估代价在整个 $f(n)$ 中所占比重使得规划的大致路径趋向于终点。其中：

$$\omega(n)=1+\left|\cos\theta_{(s,g)}-\cos\theta_{(j,g)}\right| \tag{4-14}$$

$\cos\theta_{(s,g)}$ 为起点与终点连线夹角的余弦值，$\cos\theta_{(j,g)}$ 为当前父节点搜索的临节点与终点的余弦值。

4.4.2 基于 A*算法的双向搜索优化

本研究基于 A*算法的基础进行改进。双向 A*算法与常规 A*算法的不同之处在于，在规划过程中，同时把起始点和终点作为父节点向外扩展。创建两个全局变量 closeList1、closeList2 分别存放起点向终点与终点向起点遍历的节点，当遍历到的节点靠近或者重叠时停止扩展，根据最后存放在 closeList1、closeList2 中的两个节点确定的两条路径形成最终优化后的路径。两个方向的评价函数分别如公式（4-15）、（4-16）：

$$f(n_1)=g(n_1)+\left(1+\left|\cos\theta_{(s,g)}-\cos\theta_{(j_1,g)}\right|\right)\cdot h(n_1) \tag{4-15}$$

$$f(n_2)=g(n_2)+\left(1+\left|\cos\theta_{(s,g)}-\cos\theta_{(j_2,s)}\right|\right)\cdot h(n_2) \tag{4-16}$$

其中 $\cos\theta_{(j_1,g)}$ 为正向搜索父节点的领域节点与目标点的余弦值，$\cos\theta_{(j_2,g)}$ 为反向搜索父节点的领域节点与起点的余弦值，$h(n_1)$、$h(n_2)$ 为预估代价。

当 closeList1、closeList2 中存在相同节点时，停止搜索并计算总代价值，总代价值计算公式如（4-17）：

$$f = g(n_1) + g(n_2) + \sqrt{\left(P_{x_1} - P_{x_2}\right)^2 + \left(P_{y_1} - P_{y_2}\right)^2} \tag{4-17}$$

$\left(P_{x_1}, P_{y_1}\right)$、$\left(P_{x_2}, P_{y_2}\right)$ 分别为 closeList1、closeList2 中两个相同节点的坐标。

4.4.3 基于 A*算法的节点搜索策略优化

常规 A*算法从起点开始朝着周围 8 个方向的临节点搜索。如图 4-13 所示：

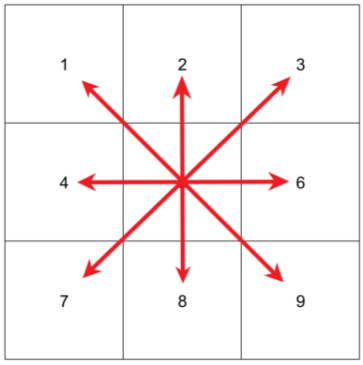
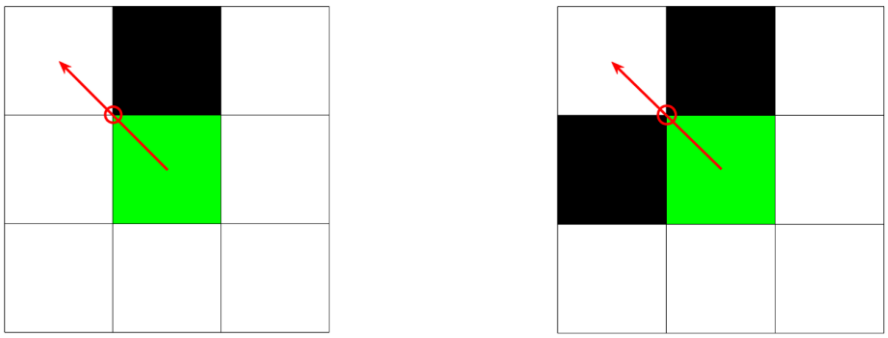


图 4-13 节点 8 方向搜索

箭头方向表示搜索方向和可移动方向；在实际应用场景中，小车可以进行横纵向移动、对角线移动，规定小车移动步长为 1 或 $\sqrt{2}$ 。

由于无人车具有一定的体积，常规的搜索策略往往只考虑了车身的中心点位置，未充分考虑车体的整体尺寸和形状，在规划行驶路径时可能会面临与障碍物发生碰撞的风险。为了确保安全有效的行驶，在算法中增加障碍物检测方法。如图 4-14 所示：



(a) 对角一边为障碍物　　(b) 对角两边都为障碍物

图 4-14 障碍物碰撞风险

在进行斜向移动时如果一侧或两侧存在障碍物，则视为此方向不可通过，需要重新规划路径。为了避免发生剐蹭、碰撞等风险，对搜索策略进行改进。如图 4-15 所示：

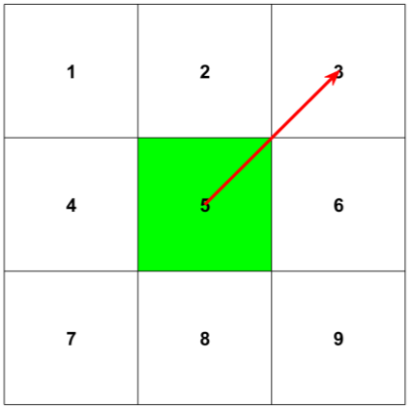


图 4-15 改进搜索策略

位置 5 为小车当前位置，对周围临近的 8 个子节点进行搜索，每当对对角线节点进行搜索时，不仅要判断其是否为障碍物，还要对其临近的两个节点进行判断，若两个临近节点都不为障碍物，则此子节点可以通过，反之则不行。例如对节点 3 进行搜索时，首先判断 3 是否为自由区域，不为则继续判断相邻节点 2 和 6 是否为自由区域，如果有一方为障碍物，则方向 3 视为不可通过。同理，其他 3 个对角线在进行判断时也采用同样的策略。

此外，为了减少冗余节点的数量，在对周围 8 个子节点进行搜索时，先对其进行正方向判断，如果满足条件，则将此节点视为可选节点，计算其预估代价，否则忽略此节点。判断当前节点是否为正方向的可选节点条件如下：

$$\sqrt{\left|x_g-x_p\right|^2+\left|y_g-y_p\right|^2}\geq\sqrt{\left|x_g-x_c\right|^2+\left|y_g-y_c\right|^2} \tag{4-18}$$

其中$\left(x_g,y_g\right)$、$\left(x_p,y_p\right)$、$\left(x_c,y_c\right)$分别为目标节点、当前节点和子节点在区域 P 的坐标。由于充电车的起点位置一般是固定已知的，而目标点取决于需要充电的车辆位置，往往是随机未知的，如果目标点向起点的搜索策略也采用此方法，可能会出现终点处于多个方向都为障碍物且无可选节点的情况，如图 4-16 所示，红色为目标点，在目标点附近有多个障碍物，而自由区域又不满足可选节点的判断条件，使得算法停滞。

因此为了避免出现无解的情况，起点向目标点搜索时采用正方向可选节点判断的策略，而终点向起点保留之前搜索策略的办法。

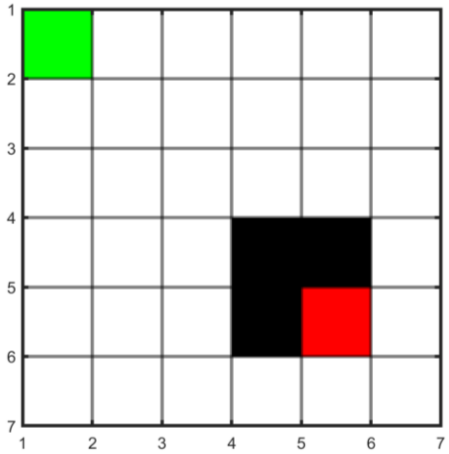


图 4-16 终点处无可选节点

4.4.4 三次 B 样条曲线路径平滑

三次 B 样条曲线在计算机图形学、CAD/CAM 设计和数据插值等领域广泛应用的曲线表示方法，具有平滑、灵活和易于控制等特点。“B”代表“Basis”（基函数），“三次”则表明该曲线的阶数，即每段曲线由四个控制点通过三次多项式组合来构成。B 样条曲线具有局部控制性，对任一控制点的调整只会影响该点附近的曲线段，能够随意修改曲线而不干扰整体结构。在每个控制点处都具有一阶和二阶导数连续性，在各个曲线区间内实现平滑过渡。

生成 B 样条曲线时，首先需要确定一组控制点，通过计算相应的基函数定义曲线形状，通过调整控制点间距改变曲线的光滑度和形状。与传统插值方法相比，更能有效避免过拟合现象。

因此本研究使用三次 B 样条曲线对规划出的路径进行平滑处理。一个 k 次 B 样条函数包含了 k+1 个基函数。这些基函数被用来描绘曲线的形状。

其定义如式（4-19）、（4-20）：

$$S_{m,k(u)}=\sum_{i=0}^{n}P_{i+m}N_{i,k(u)} \tag{4-19}$$

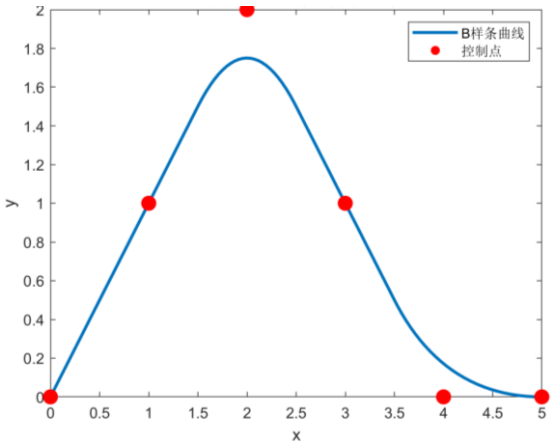
$$N_{i,k(u)}=\frac{1}{k!}\sum_{j=0}^{k-i}(-1)^{j}\,C_{k+1}^{j}\left(u+k-i-j\right)^{k}\qquad u\in[0,1],i=0,1,\ldots,k \tag{4-20}$$

其中，$S_{m,k(u)}$ 是在参数 u 处的曲线位置，P_{i+m} 是控制点，$N_{i,k(u)}$ 是 k 阶 B 样条基函数。当 k=3 时，

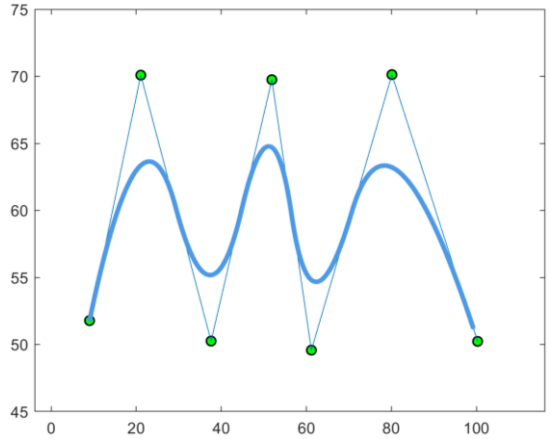
$$\begin{cases} N_{0,3(u)} = \dfrac{1}{6}\left(-u^3 + 3u^2 - 3u + 1\right) \\ N_{1,3(u)} = \dfrac{1}{6}\left(3u^3 - 6u^2 + 4\right) \\ N_{2,3(u)} = \dfrac{1}{6}\left(-3u^3 + 3u^2 + 3u + 1\right) \\ N_{3,3(u)} = \dfrac{1}{6}u^3 \end{cases} \quad u \in [0,1] \tag{4-21}$$

这四个基函数的线性组合可以用来构建整个三次 B 样条函数。

将每个基函数设置在控制点上，并根据相应的权重对它们进行线性组合，以得到最终的 B 样条函数。这些权重可以通过插值或逼近技术获得，并且它们决定了曲线在每个控制点处的形状。如图 4-17(a)和 4-17(b)所示分别为 6 个控制点与 7 个控制点的 3 次 B 样条曲线，可见尽管控制点数量不同，其生成的曲线都能保持高度的平滑性。



(a) 6 个控制点



(a) 7 个控制点

图 4-17　3 次 B 样条曲线

4.4.5 改进 A*算法具体流程

（1）初始化参数：包括起点、终点、环境地图等。创建栅格环境模型：将环境地图转换为栅格化的表示方式，便于进行搜索。

（2）创建 4 个列表，分别为 openList1、closeList1、openList2、closeList2。openList1、openList2 分别存储正向搜索和反向搜索中待调查的节点，closeList1、closeList2 存储已经调查的节点。其中，openList1 和 openList2 分别初始化为起点和目标点，closeList1 和 closeList2 初始化为空列表。

（3）正向搜索从 openList1 中选择代价值最小的点作为下一次循环的父节点，将其加入到 closeList1 中，并将其从 openList1 中剔除；反向搜索同理不再赘述。

（4）正向搜索判断当前父节点是否为目标点，如果是则搜索过程结束，生成规划后的路径；如果不是则按照如下搜索策略进行调查（反向搜索同理）：选取正方向的可选节点(4.4.3 节描述的搜索策略)；忽略周围为障碍物的节点；调查未存在于 closeList1 和 openList1 中的节点；

反向搜索为防止两个方向的搜索同时陷入局部最优，决定按照常规搜索策略进行。判断当前父节点是否为目标点，如果是则搜索过程结束，生成规划后的路径；如果不是则继续对其领域的 8 个子节点进行调查

（5）正向搜索对此轮遍历的剩余节点进行判断，重新计算已经存在于 openList1 中的节点的代价值 $f\left(n_1\right)$,并与原值进行比较，保留代价值较小的数并更新路径；若不存在于 openList1，则将其添加到 openList1 中去，计算代价值更新路径；反向搜索同理。

（6）返回第三步进行循环搜索，直到 closeList1、closeList2 中存在相同节点时停止搜索。

（7）使用三次 B 样条曲线对规划出的路径进行平滑处理。

改进后的 A*算法具体流程图如图 4-18 所示：

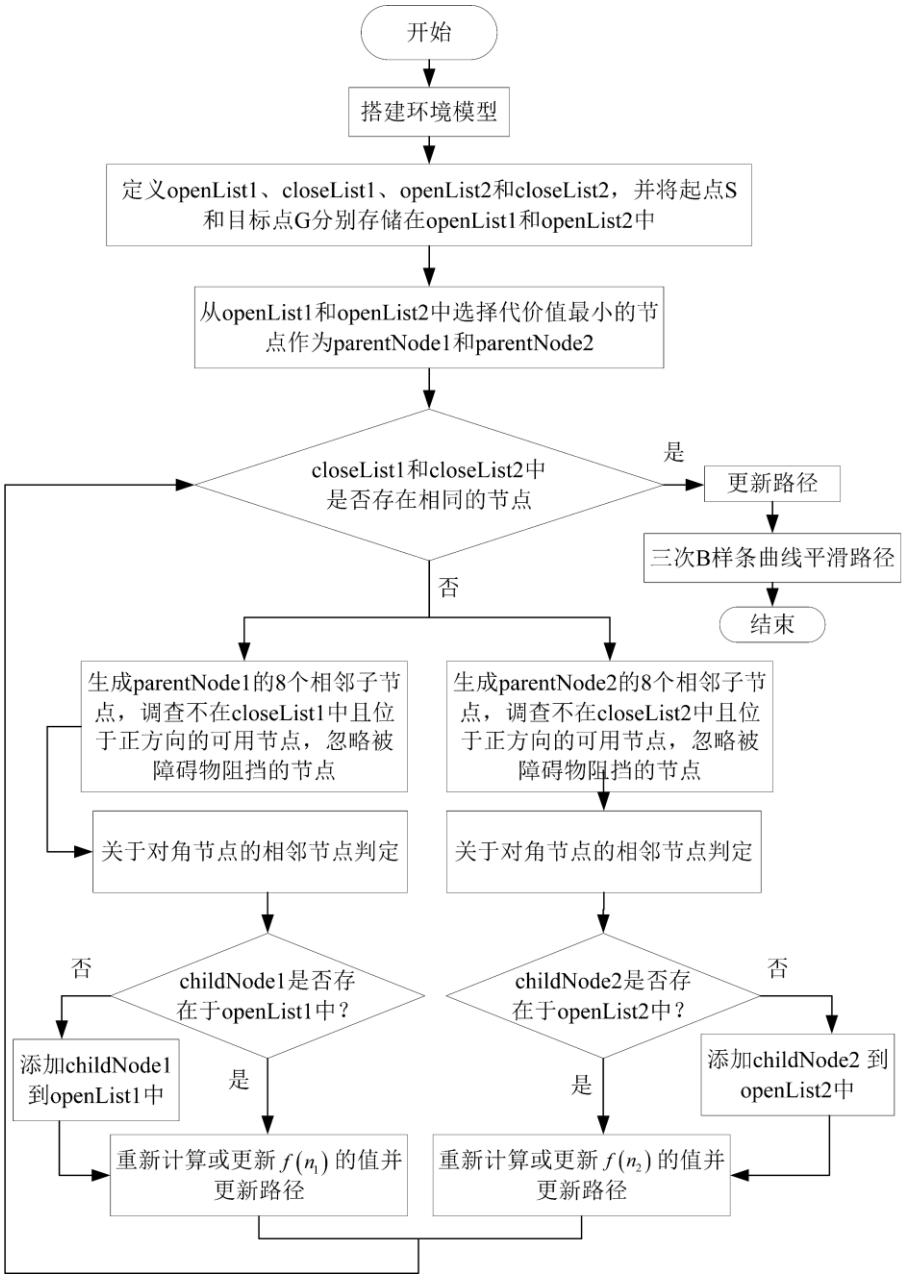


图 4-18 改进 A*算法流程图

4.5 仿真结果与分析

本章使用 matlab2023a 对以上描述的理论进行实验验证，并对结果进行分析。首先为了验证算法的有效性，在栅格环境为 30*30、障碍物占比为 1/3 的地图上对 Dijkstra 算法、RRT 算法、双向 A*算法以及改进后的 A*算法进行实验对比；然后在栅格环境为 30*40、障碍物占比约为 1/3 的模拟应用场景，选取左上角起点、右下角目标点以及左上角起点、狭窄处的目标点两组进行仿真实验；为了减

少实验结果的误差，分别对每个算法进行 20 次实验并统计每轮实验下的各个数据，最后取平均值，实验结果如图 4-19 至 4-30。

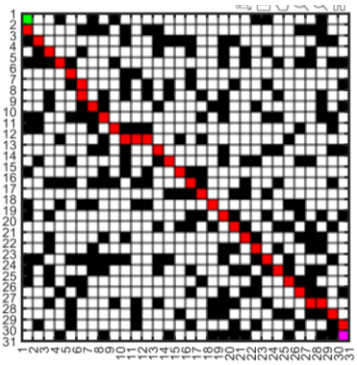


图 4-19 Dijkstra 规划路径（30*30;1/3）

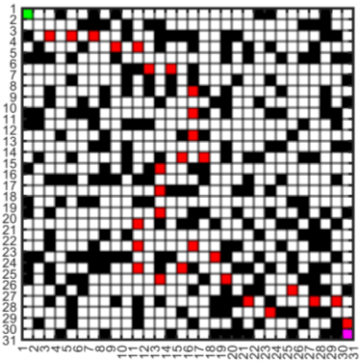


图 4-20 RRT 规划路径（30*30;1/3）

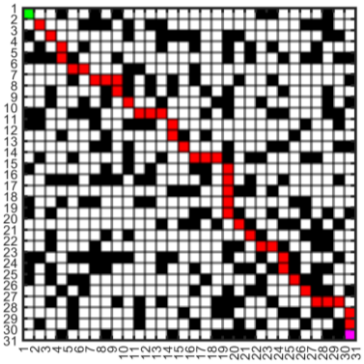


图 4-21 双向 A*规划路径（30*30;1/3）

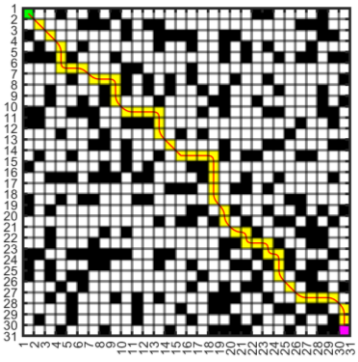


图 4-22 改进 A*算法规划路径（30*30;1/3）

表 4-1 算法实验结果统计（30*30;1/3）

算法	搜寻节点数量	路径的栅格数量	路径长度	搜寻时间（s）	是否规避障碍物
Dijkstra	741	46	52.8701	0.3152	N
RRT	244	56	73.3553	0.2441	N
双向A*算法	415	47	54.8701	0.3905	N
改进A*算法	297	50	58.4558	0.1815	Y

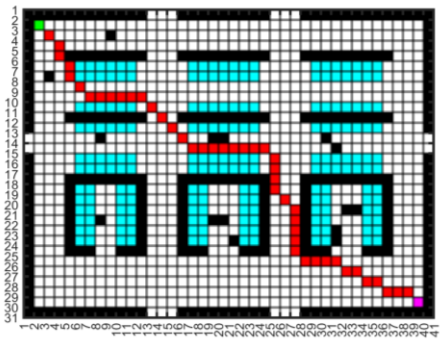


图 4-23 Dijkstra 规划路径（30*40）

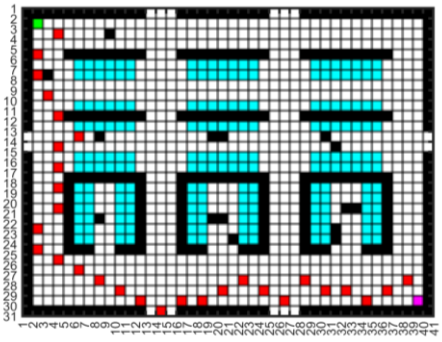


图 4-24 RRT 规划路径（30*40）

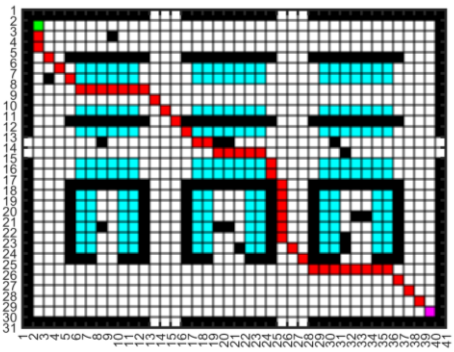


图 4-25 双向 A*算法规划路径（30*40）

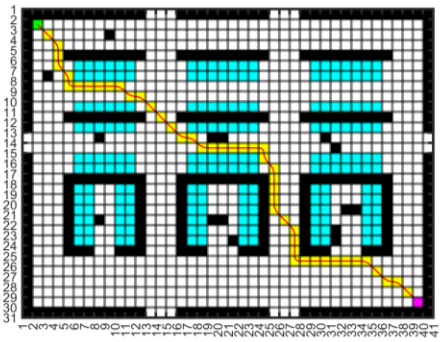


图 4-26 改进 A*算法规划路径（30*40）

表 4-1 算法实验结果统计（30*40）

算法	搜寻节点数量	路径的网格数量	路径长度	搜寻时间（s）	是否规避障碍物
Dijkstra	642	33	42.7641	0.5001	N
RRT	222	46	58.1781	0.4431	N
双向A*	363	40	46.8701	0.3242	N
改进A*	179	48	45.1421	0.2417	Y

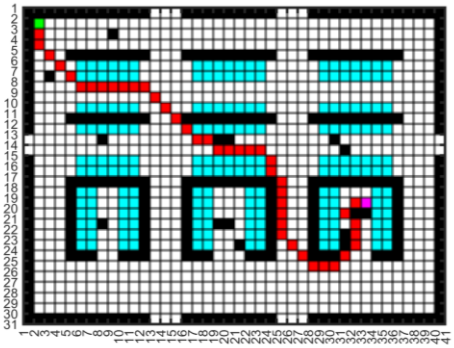


图 4-27 Dijkstra 算法（室内目标点）

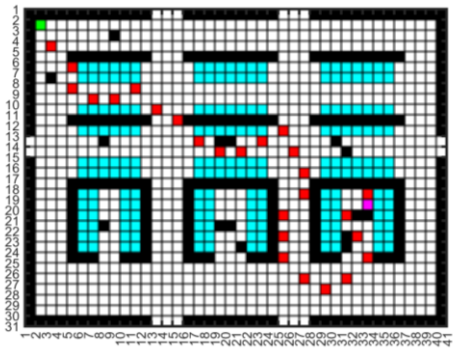


图 4-28 RRT 算法（室内目标点）

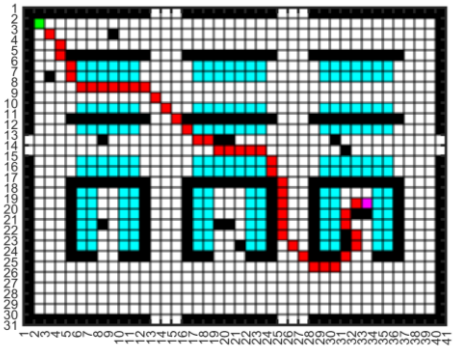


图 4-29 双向 A*算法（室内目标点）

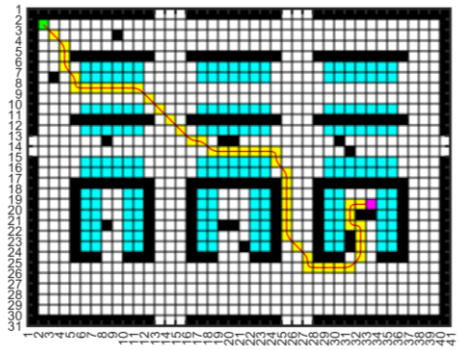


图 4-30 改进 A*算法（室内目标点）

表 4-2 算法实验结果统计（室内目标点）

算法	搜寻节点数量	路径的网格数量	路径长度	搜寻时间（s）	是否规避障碍物
Dijkstra	742	44	52.8701	0.3146	N
RRT	265	58	66.6940	0.4431	N
双向A*	204	45	51.7526	0.3671	N
改进A*	228	50	54.3848	0.1779	Y

根据以上实验结果可知：

在环境为 30*30 的栅格仿真实验中，四种算法都能够规划出到达目标点的路径，验证了算法的有效性；并且 Dijkstra 算法、双向 A*算法、改进 A*算法在每一次实验中规划出的路径、长度不变。改进后的 A*算法相比于双向 A*算法规划时间降低了 53.52%、搜索节点数量减少了 28.43%。在环境为 30*40 的栅格地图实验中，改进后的 A*算法在进行斜向移动时会主动判断静态障碍物并进行避障，虽然路径长度相比于 Dijkstra 算法以及双向 A*算法有略微增加，但其路径更适合于无人充电车的作业行驶路径，提供了产品的安全保障。由表二和表三两组仿真实验结果可知，改进后的 A*算法在规划时间、路径平滑度等方面都具有优势，相比于双向 A*算法规划时间分别降低了 25.45%、51.54%。其不仅减少了节点资源的浪费、提高了算法的搜索效率，还对规划后的路径进行平滑处理，输出一条适合机器人安全作业的可行路径。

4.6 本章小结

本章基于双向 A*算法的基础上对其进行改进，通过优化 A*算法的启发式函数、正方向可选节点引导，并对子节点进一步考察，最后使用 B 样条曲线对路径进行平滑处理，在不同环境下的实验结果表明，改进后的 A*算法在复杂环境下以及不同位置目标点时都能规划出一条安全可靠的路径，并提高了作业效率、减少了不必要节点的收录，验证了算法的可行性、有效性以及优越性。

第 5 章 基于 ROS 平台的自动驾驶实验

5.1 引言

本研究在 Ubuntu 20.04 环境下构建基于 ROS 的智能导航系统，采用遥控和自动导航双模式进行测试。使用 SLAM 算法实现同步定位与地图构建。为了测试安全保障，机器人本身具有急停按钮，此外通过改进 A*算法与动态窗口法（DWA）的协同机制完成导航任务。

5.2 基于 SLAM 与改进 A*算法自动驾驶测试

本章基于中科深谷的室外重载作业驱控一体底盘进行测试，测试地点位于安徽省皖西学院机电楼 407 实验室。安装在车辆前端的思岚 A2 激光雷达，通过 ROS 驱动包订阅相应话题即可获取激光雷达的数据信息。机器人底盘及试验场地如图 5-1 和 5-2 所示：

图 5-1 对角一边为障碍物

图 5-2 对角两边都为障碍物

首先打开机器人底盘总开关，此时小车内部会自动启动手柄控制节点，将手柄与小车连接即可控制底盘进行移动。使用虚拟机 VMware 的 Ubuntu20.04 系统远程连接机器人的内部电脑显示器，安装 gmapping 等必要的 ROS 包，为后续操作做好准备。设置坐标系之间的 TF 变换关系，建立/odom 和/base_link 之间的联系；将/odom 坐标系的位姿作为 move_base 的输入；配置相关的 move_base.xml 文件，包括全局和局部代价地图等设置；为了优化 URDF 模型，解决模型冗长、内容重复和参数修改繁琐的问题，决定使用 XACRO 格式进行改进。

在 ROS 平台中，为完整进行建图和导航，需要创建并订阅所需话题。激光 SLAM 算法需要涉及三个部分的内容：话题订阅、话题发布和服务。

在话题订阅中，分别需要订阅 tf、scan、map 话题，tf 话题可实时获取并记录激光雷达、底盘相对于某个基准坐标系（如世界坐标系或机器人自身坐标系）的位姿关系；scan 话题由激光雷达节点发布，包含了激光雷达扫描到的环境数据，包含了扫描点的距离、角度等信息；scan 话题可以获取到激光雷达扫描到的环境信息，为后续的环境感知与地图构建提供关键数据；map 话题包含了环境的 meta 数据，如占据网格（occupancy grid）描述了环境中不同位置的占据情况（被占据、空闲或未知）。通过订阅 map 话题，可以获取到完整的地图信息，并在 RVIZ 等可视化工具中进行展示，实现地图的构建、保存与加载等操作。

在完成激光 SLAM 算法所需要的相关话题及服务后，就可以在 ROS 中进行建图的相关工作了。

5.3 实验结果

在 ROS 终端使用指令 roslaunch zeus_s2_slam gmapping_base.launch 启动 Launch 文件进行建图，如图 5-3 和 5-4 所示：

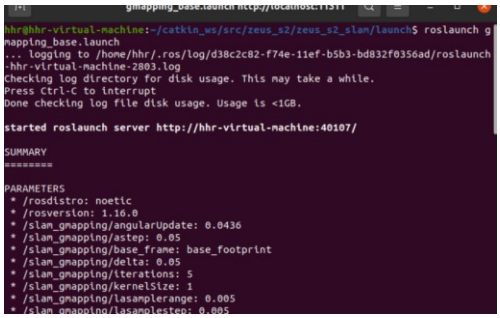


图 5-3 Launch 启动建图指令

图 5-4 机器人建图

通过手柄控制机器人移动，在实验室内进行探索来构建地图。在这个过程中，可以使用 RVIZ 添加 add map 并订阅/map 话题，实现实时地图构建过程的可视化，如图 5-5 和 5-6 所示。

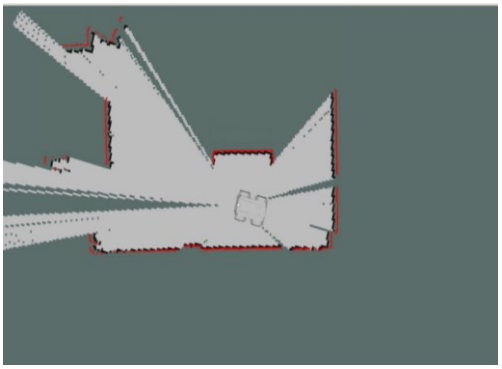


图 5-5 RVIZ 可视化 SLAM 建图

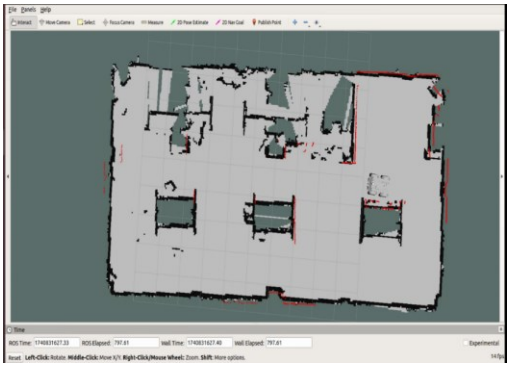


图 5-6 建图工作完成

建图完成后，通过 map_server 节点，将地图保存在本地。如图 5-8 所示。

图 5-7 完成建图工作

图 5-8 保存后的地图

接着在 ROS 中进行导航工作。打开机器人底盘电源开关并用电脑远程连接机器人，将已生成的地图 YAML 配置文件存放于工作空间源代码（src）目录之下的 map 文件夹中，随后在负责启动导航的 launch 文件中配置地图文件和调用负责导航的 nav 功能包，再于终端中执行 launch 文件即可开始自主导航功能。在 RVIZ 环境中订阅地图数据/map 即可开始呈现完整的地图信息，如图 5-9 所示为准备开始导航工作的机器人。

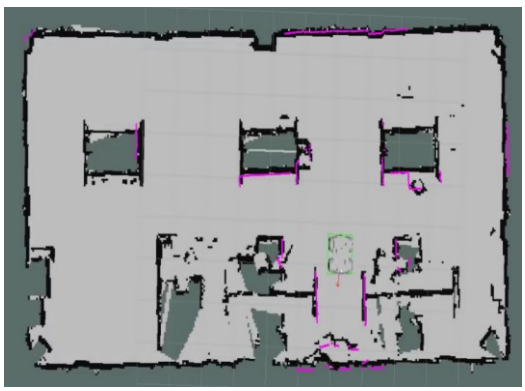
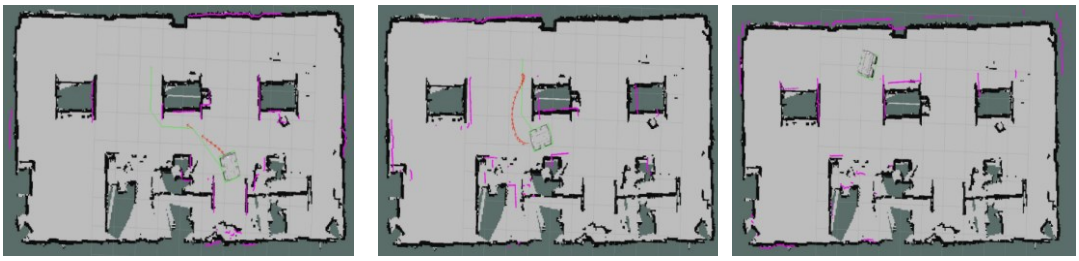


图 5-9 订阅地图准备导航工作

在 Launch 启动文件中添加全局规划器和局部规划器分别调用 A*算法和 DWA 算法后，通过 RVIZ 中的 2D Nav Goal 点击地图上的点即可设定目标点，系统会规划出一条可行的大致路径，如图中的绿色实线所示，小车会逐渐按照绿色实线缓慢移动；当遇到局部障碍物或动态障碍物时，局部规划器通过 DWA 算法实现实时避障，小车会根据规划出的局部路径绕开障碍物继续向目标点前进，直至抵达终点，如图 5-10 所示。



(a) 路径规划完成　　(b) 前往目标点　　(c) 抵达终点

图 5-10 RVIZ 导航过程图

在使用 2D Nav Goal 选中目标点后，小车快速做出了决策，规划出一条通往目标点的最优全局路径，并在拐角处有效避开障碍物，同时在 DWA 局部路径规划算法的辅助下实现实时避障，最终安全抵达了终点。

5.4 本章小结

本章围绕基于 ROS 的智能导航系统设计与实现展开研究，通过理论分析、算法改进和实验验证完成整个论文工作流程。

在 Ubuntu 20.04 的 ROS 环境下搭建了软硬件协同的智能导航系统，采用模块化设计，实现了遥控、自主导航机器人控制方案。在 ROS 框架下完成坐标变换树、代价地图等核心模块配置，通过订阅/scan、/map 等话题实现实时环境感知。研究实现了环境地图的构建，通过改进 A*全局路径规划与 DWA 算法的融合实现实时局部避障。设计全局规划和局部规划协同工作：A*算法生成全局最优路径后，DWA 算法动态修正局部轨迹，解决静态地图与动态障碍物的路径冲突。

第 6 章 总结与展望

6.1 总结

本文以移动充电机器人为研究对象，针对其在复杂环境中的自主导航需求，围绕点云配准与路径规划算法的优化展开系统性研究。通过构建基于 ROS 的软硬件协同仿真平台，提出改进的 ICP 点云配准方法与 A*路径规划算法，使用 MATLAB 进行仿真测试，并结合物理仿真实验验证算法有效性，对实现在复杂场景中的自动驾驶有重要的意义。

主要研究内容包括机器人仿真环境和平台的搭建、对激光雷达点云配准算法和全局路径规划算法的仿真及优化、机器人底盘的实际测试；针对传统 ICP 算法的局限性，结合 KD-Tree 加速搜索与离散点祛除策略，有效提升了配准精度、加快了配准效率。通过对 NDT、A*、DIJKSTRA 等算法的综合对比分析，对常规的 A*算法进行优化，有效提高了算法的规划效率和质量；基于 ROS 操作系统在实验场地实现了机器人的自主驾驶。本文主要工作内容如下：

（1）首先对国内外有关充电机器人的发展历程和相关研究做了综述，围绕新能源技术快速发展对智能充电机器人的迫切需求，针对复杂环境中机器人的自主导航问题开展研究，阐述了点云配准算法与路径规划算法的国内外研究现状，提出如今算法存在的不足，针对实际应用场景进行优化。

（2）基于 ROS 系统性地完成了仿真环境与机器人模型的搭建；通过 URDF 模型完成了仿真小车的三维模型搭建，定义了机器人关节、连杆及传感器的几何结构与物理属性，基于 Gazebo 将 URDF 模型与物理引擎结合，搭建了包含障碍物、行人等仿真环境，为后续点云数据采集、点云配准算法的优化奠定基础。

（3）提出了一种基于 ISS、RANSAC 算法和改进 ICP 算法相结合的点云配准方法，并通过仿真实验验证了其有效性。首先，基于 Gazebo 搭建的多障碍物仿真环境和机器人、传感器模型，完成了点云数据的动态采集。粗配准初步对齐点云位姿；在精配准阶段，结合传统 ICP 算法框架，设定阈值半径祛除离散点，通过采集数据与斯坦福大学的公开数据集的仿真实验结果表明，与传统 ICP 算法相比，本文算法在配准精度与算法收敛速度都有效提升。

（4）针对传统 A*算法在复杂场景中路径规划效率低、节点资源浪费率较高及路径平滑性不足的问题进行优化，并通过仿真栅格地图验证有效性。通过重构启发式函数、对角线障碍物判定与双向搜索策略；结合正方向引导机制减少无效节点搜索规模；采用 B 样条曲线对规划路径进行平滑处理，提升无人充电车行驶轨迹的连续性与安全性。实验表明，改进算法相较于双向 A*算法，规划效率在两组实验中分别缩短了 25.45%、53.52%；在某些复杂情况下尽管路径长度略有增加，但其生成的路径曲率平缓、与障碍物保持安全间距，为机器人行驶过程提供安全保障，更契合机器人的作业场景需求。

（5）在 Ubuntu 20.04 环境下基于 ROS 搭建了智能导航系统，模块化设计机器人内部结构，包括建图、定位、路径规划与避障功能。通过手柄控制机器人运动扫描整个实验场地完成 SLAM 建图；导航系统通过订阅/map、/scan 等话题实时感知环境，利用 TF 树与代价地图确定机器人位姿，结合改进后的 A*算法与 DWA 算法共同完成机器人底盘的导航功能成功，实现了实验室场景下的地图构建与自主导航。

6.2 展望

本文在充电机器人自主导航、点云配准与路径规划等方向进行了一定研究，但在实际工程应用与算法优化方面仍需进一步探索。包括以下几方面：

（1）当前 ICP 算法在配准时仍需要较好的点云初始位置，如何提高粗配准算法的配准精度、优化全配准流程有待研究。同时算法对更复杂的环境可能会有不同的敏感度，后续工作可以探究不同噪声环境对整体配准稳定性的影响。

（2）当前路径规划算法在动态环境中的避障实时性仍然不足，对局部路径规划算法的进一步研究具有重要意义，在面对不同路况中都能及时获取最优路径。

（3）当前实物测试实验中只包含了路径规划算法的测试，缺少了点云配准算法对建图效果的实际验证，计划在后续工作中继续深入学习，探究点云配准算法对实际建图质量的影响。理论学习应当与工程实践紧密结合，继续加强对实际工程产品、软硬件结合的闭环学习。

参考文献

[1] 黄蔚. CKF 及鲁棒滤波在飞行器姿态估计中的应用研究[D]. 黑龙江:哈尔滨工程大学, 2015.

[2] 熊有伦. 机器人学:基础理论与应用实践[M]. 黑龙江:哈尔滨工业大学出版社, 2020: 3.

[3] 夏希品. 中国机器人产业发展报告 2019 发布[J]. 传感器世界, 2019(25): 42-43.

[4] 王晓芸,崔培,陈晓.轮式移动机器人文献综述[J].石家庄铁路职业技术学院学报,2019,18(02):66-70.

[5] 龚志力, 谷玉海, 朱腾腾, 任斌. 一种基于机器人操作系统的代价地图自适应膨胀半径设置方法[J]. 科学技术与工程, 2021, 21(09): 3662-3668.

[6] Zhu S S, Zhao P, et al. Design of Piston Rod Inspection Robot for Hydraulic Hoist based on Vision[J]. 国际设备工程与管理 (英文版), 2022, 27(1):1-17.

[7] Duan Y, Zheng Y, Lu J, et al. Structural Relational Reasoning of Point Clouds[C]. IEEE/CVF Conference on Computer Vision & Pattern Recognition. IEEE, 2019.

[8] Cai Z, Chin T J, Bustos A P, et al. Practical optimal registration of terrestrial LiDAR scan pairs[J]. ISPRS Journal of Photogrammetry and Remote Sensing, 2019, 147(JAN.):118-131.

[9] Myronenko A, Song X. Point Set Registration: Coherent Point Drift[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2010, 32(12):2262-2275.

[10] Jiang H, Dang Z, Wei Z, et al. Robust Outlier Rejection for 3D Registration with Variational Bayes[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 1148-1157.

[11] 刘铮, 杨隆, 江昱晓, 吕小荣. 国内外农业四轮机器人的发展现状及展望[J]. 农业科学, 2022, 12(9): 890-897.

[12] Behringer R. The DARPA grand challenge - autonomous ground vehicles in the desert[J]. IFAC Proceedings Volumes, 2004, 37: 904-909.

[13] Okunevich I, Trinitatova D, Kopanev P, et al. DeltaCharger: Charging Robot with Inverted Delta Mechanism and CNN-driven High Fidelity Tactile Perception for

Precise 3D Positioning[J]. 2021, 6(4): 7604-7610.

[14]Hoss Design USA And Car Charging Bot Join Forces To Revolutionize Product Design With Wireless Charging Robot[J]. M2 Presswire, 2023.

[15]Altamirano C M, Rakhmatulin Viktor, FedoseevAleksey, et al. OmniCharger: CNN-Based Hand Gesture Interface to Operate an Electric Car Charging Robot through Teleconference[J]. ACM Transactions on Human-Robot Interaction, 2024，13, 3, Article 33 (September 2024), 18 pages.

[16] Mark Moran. Hyundai builds an EV charging robot[J]. Parking Review, 2023, (TN.370): 11a.

[17] Zhang H, Zhu W, Huang Y. A research on the control strategy of automatic charging robot for electric vehicles based on impedance control[J]. Journal of Physics: Conference Series, 2022, 2303.

[18]宋凯, 朱春波, 李阳, 等. 基于磁耦合谐振的自主无线充电机器人系统设计[J]. 电工技术学报, 2014, 29(09): 38-43.

[19]Ren Y, Zeng H, Liang Y. SFE-SLAM: an effective LiDAR SLAM based on step-by-step feature extraction[J]. Applied Intelligence,2024,55(2): 87-87.

[20]李枭凯, 李广云, 索世恒, 等. 激光 SLAM 技术进展[J]. 导航定位学报, 2023, 11(4): 8-17.

[21]衣明悦. 基于激光雷达的点云配准算法综述[J]. 汽车实用技术, 2021, 46(17): 212-214.

[22]张晓, 张爱武, 王致华. 基于改进正态分布变换算法的点云配准[J]. 激光与光电子学进展, 2014, 51(4): 041002.

[23]NGUYEN T, 刘修国, 王红平, 等. 基于激光扫描技术的三维模型重建[J]. 激光与光电子学进展, 2011, 48(8): 081201.

[24]张靖, 于伶. 结合局部特征匹配和点云配准的姿态估计网络[J]. 重庆工商大学学报(自然科学版), 1-9[2024-10-20].

[25]蓝秦隆. 基于改进 GA-SA 的点云配准算法研究[D]. 武汉武汉大学, 2021.

[26]胡加涛, 吴晓红, 何小海, 等. 一种基于几何特征由粗到细点云配准算法[J]. 科学技术与工程, 2020, 20(5): 1947-1952.

[27] Sun Ruiyang, Zhang Enzhong, Mu Deqiang, et al. Optimization of the 3D point

cloud registration algorithm based on FPFH features[J]. Applied Sciences, 2023, 13(5): 3096.

[28] Wang C, Xiong X, Zhang X, et al. A 3D Point Cloud Feature Identification Method Based on Improved Point Feature Histogram Descriptor[J]. Electronics, 2023, 12(17): 3736-.

[29] 许汉文, 高文根, 王瑞, 等. 结合组合特征点的改进 ICP 点云配准方法[J/OL]. 重庆工商大学学报(自然科学版), 1-10[2024-10-20].

[30] Zhang Jing, Yu K, Wen Zheng, et al. 3D reconstruction for motion blurred images using deep learning-based intelligent systems[J]. Computers, Materials & Continua, 2021(2): 2087-2104.

[31] Mellado N, Aiger D, Mitra N J. Super4PCS: fast global pointcloud registration via smart indexing[J]. Computer Graphics Forum, 2015, 33(5): 205-215.

[32] Xu Guangxuan, Pang Yajun, Bai Zhenxu, et al. A fast point clouds registration algorithm for laser scanners. Applied Sciences. 2021, 11(8): 3426.

[33] Biber P, Strasser W. The normal distributions transform: a new approach to laser scan matching[J]. Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453), Las Vegas, NV, USA, 2003, pp. 2743-2748.

[34] Magnusson M. 3D scan matching for mobile robots with application to mine mapping[J]. Örebro University, 2006.

[35] Besl P J, Mckay H D. A method for registration of 3-D shapes[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 1992, 14(2): 239-256.

[36] Guan Wei, Li Wentao, Ren Yan. Point cloud registration based on improved ICP algorithm[C]. 2018 Chinese Control And Decision Conference (CCDC), Shenyang, China, 2018: 1461-1465.

[37] 许哲, 董林啸, 吴家跃. 改进 ICP 算法的激光雷达点云配准[J]. 测绘通报, 2024(4): 1-5.

[38] 张赵良, 董一鸣, 朱菊香, 等. 基于 ISS 特征点结合改进 ICP 的点云配准算法[J]. 应用激光, 2023, 43(06): 124-131.

[39] Chen Songlin, Nan Liangliang, Xia Renbo, et al. PLADE: a plane-based descriptor

for point cloud registration with small overlap[J]. IEEE Transactions on Geoscience and Remote Sensing, 2020, 58(4): 2530-2540.

[40] Bai X, Luo Z, Zhou L, et al. Pointdsc: Robust point cloud registration using deep spatial consistency[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 15859-15869.

[41] Tsin Y, Kanade T. (2004). A Correlation-Based Approach to Robust Point Set Registration[J]. ECCV. 3. 558-569.

[42] Myronenko A, Song X. Point Set Registration: Coherent Point Drift[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2010, 32(12):2262-2275.

[43] Yang H, Shi J, Carlone L. TEASER: Fast and Certifiable Point Cloud Registration[J]. IEEE, 2021(2).

[44] Liu Fu, Chen Hongming. Research on Path Planning Method of Unmanned Vehicle Based on RRT Algorithm[J]. Machinery Design and ManufacturingEngineering, 2020, 19(8) : 109-112.

[45] Dijkstra, E.W. A note on two problems in connexion with graphs[J]. Numer. Math. 1, 269-271 (1959).

[46] Yin W, Yang X. A Totally Astar-based Multi-path Algorithm for the Recognition of Reasonable Route Sets in Vehicle Navigation Systems[J]. Procedia-Social and Behavioral Sciences, 2013, 96: 1069-1078.

[47] Stentz A. Optimal and efficient path planning for partially-known environments[J]. Proceedings of the 1994 IEEE International Conference on Robotics and Automation, Vol. 4, 1994, pp. 3310-3317.

[48] Lyu Desheng, Chen Ziwei, Cai Zesu, et al. Robot path planning by leveraging the graph-encoded Floyd algorithm, Future Generation Computer Systems[J]. Volume 122, 2021, Pages 204-208.

[49] Ye Lei, Wu Fengyun, Zou Xiangjun, et al. Path planning for mobile robots in unstructured orchard environments: An improved kinematically constrained bi-directional RRT approach[J]. Computers and Electronics in Agriculture, Volume 215, 2023, 108453, ISSN 0168-1699.

[50] Yue L, Chen H. Unmanned vehicle path planning using a novel ant colony

algorithm[J]. EURASIP Journal on Wireless Communications and Networking, 2019, 2019: 1-9.

[51]王维, 裴东, 冯璋. 改进 A*算法的移动机器人最短路径规划[J]. 计算机应用, 2018, 38(5): 1523-1526.

[52]李云龙, 梁波, 薛新华, 等. 基于双向 A*算法的路径规划问题研究[J]. 电子质量, 2023(2): 1-4.

[53]Yux, Lianghw, Dumb, et al. An improved A* algorithm for searching infinite neighborhoods [J]. Robot, 2014, 36(5): 627-633.

[54]张红梅, 李明龙, 杨乐. 基于改进Ａ*算法的移动机器人安全路径规划[J]. 计算机仿真，2018, 35(4): 319-324.

[55]李森杰，郑洪瀛，杨超，等. 基于 A* 与 DWA 算法的混合路径规划算法研究[J]. 吉林大学学报(信息科学版), 2022, 40(1):132-141.

[56]Rini W, Cin C, Suhartono, et al. TransTrip: A Shortest Path Finding Application for Jakarta Public Transportation using Dijkstra Algorithm[J]. Journal of computer sciences, 2018, 14(7): 939-944.

[57]Cui X L，Wang C Q，Xiong Y，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: 108246.

[58]Zhang Ly, Hany, Jang B. Research on path planning method of unmanned boat based on improved DWA algorithm[J]. Journal of Sensors, 2022. 2022: 9315483.

[59]Chen I X, Y I L. Dynamic path planning for mobile robots based on the improved A-star algorithm [J]. Academic Journal of Computing Information Science, 2021, 4(8): 73-77.

[60]王子静, 陈熙源. 基于改进 A*和 DWA 的无人艇路径规划算法[J]. 传感技术学报, 2021, 34(02): 249-254.

攻读学位期间发表的学术成果

（1）H. Hu, J. Fang, H. Bao, et al. Research on Path Planning for Unmanned Charging Vehicles based on Improved A* Algorithm. 2024 China Automation Congress (CAC), Qingdao, China, 2024, pp. 37-43.

（2）Jie Fang, Haoran Hu, Huifang Bao, et al. Research on Point Cloud Registration Algorithm for Autonomous Vehicles Based on Discrete Points Removal. (已录用)

致 谢

要感谢的人很多，但真正提笔时反而有些局促。

三年前儿童节那天收到了来自安徽工程大学的研究生录取通知书，像是上天送给我的礼物，也是自己送给自己的礼物，笨拙又珍贵，如今到了验收成果的这一天。科研之路布满荆棘，却因良师益友的陪伴与支持充满温度，谨以诚挚的谢意，献给所有助我前行的人。

向我的导师方杰教授致以深切的感激。他以渊博的学识、严谨的治学态度为我指明研究方向，无论是学术还是工程实践方面都给予耐心指导；在开题、中期到答辩各方面都能做到严格要求、一丝不苟；在企业实习的时候，多次叮嘱翁志远老师多督促我和李鸿飞的学习进度；安排鲍慧芳和张进思老师辅导我遇到的学术问题；每次出门在外都会叮嘱我“路上注意安全”，他就像我的家人一样对我的生活给予悉心的关照。多有压力才有动力，很多时候会想能够成为您的学生，是我读研路上最大的幸运。

二十余载求学路上，我的父母方玉女士和胡新文同志，他们始终是我最坚实的后盾，虽然学历不高，可教会我的道理要比书本上实在的多，以前总觉得唠叨，但当自己遇到难处了，第一个想到的还是他们。听我妈妈说她念大学那会儿，女孩子的日常生活用品花销是比男生多一点的，而姥姥由于考虑不周到，每个月只有那一点捉襟见肘的生活费，导致经常入不敷出。因为自己淋过雨所以怕我也淋着，现在条件好起来了，听我姐姐说她经常担心给我的生活费不够，就像她读书那会一样，所以每个月都会多给我转一千块钱。父亲生病后更是凭一己之力撑起整个家庭。尽管有时会和家里闹不愉快、发生口角，但这份致谢成了不善言辞的我向你们倾诉感激之情的宝贵机会。

感谢实验室的各位老师在论文阅读、设备使用、数据分析方法中提出的宝贵意见和指导，还有一同前往皖西学院学习的潘静、李鸿飞、王溯和已经毕业的李田利师姐，大家一起营造出的融洽的学术氛围。还有从头到尾充满活力、欢乐超级加倍的室友岳君豪、桑建斌、杨思远同学。

回忆起那个想要读研的起点，我最想感谢的是位名叫解蕾的女生，大学曾一度想摆烂的我成绩并不是很突出，是当时作为女朋友的你，督促着我好好学习、

积极参加比赛去争取奖学金、鼓励我考取四六级和教师资格证，携手与我共同进步，更是陪伴我一起度过了艰难的考研之路和宝贵的大学 4 年青春。与你相识之后我体会到了恋人的被爱与被关心，更学会了如何去爱别人，做一个共情、善良的人。希望我 25 年的夏天能够顺利毕业，如愿捧起学位证书，做我喜欢做的事；也祝你的研究生生活能在知识的海洋里乘风破浪，让每篇论文都在学术星河里闪耀。期待未来我们能携手书写属于我们的璀璨篇章，共赴山海。

最后，谨向参与论文评审与答辩的各位专家致以崇高敬意。文虽终而意无穷，路虽远而行将至。我将带着这份满载期望的答卷，继续在学术道路上砥砺前行。