

中国科学技术大学

硕士学位论文



面向园区的移动机器人多层级导航系统 研究设计

作者姓名： 陈亮

学科专业： 控制工程

导师姓名： 陈宗海教授 项俊平高级工程师

完成时间： 二〇二三年五月三十日

University of Science and Technology of China
A dissertation for master's degree



**Research design of multi-level
navigation system for intelligent
unmanned vehicles in the park**

Author: Chen Liang

Speciality: Control Engineering

Supervisors: Prof. Chen Zonghai, Prof. Xiang Junping

Finished time: May 30, 2023

中国科学技术大学学位论文原创性声明

本人声明所呈交的学位论文，是本人在导师指导下进行研究工作所取得的成果。除已特别加以标注和致谢的地方外，论文中不包含任何他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的贡献均已在论文中作了明确的说明。

作者签名：_____

签字日期：_____

中国科学技术大学学位论文授权使用声明

作为申请学位的条件之一，学位论文著作权拥有者授权中国科学技术大学拥有学位论文的部分使用权，即：学校有权按有关规定向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅，可以将学位论文编入《中国学位论文全文数据库》等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、汇编学位论文。本人提交的电子文档的内容和纸质论文的内容相一致。

控阅的学位论文在解密后也遵守此规定。

公开 控阅（____ 年）

作者签名：_____

导师签名：_____

签字日期：_____

签字日期：_____

摘要

移动机器人导航系统是机器人技术领域的重要研究方向之一，其核心目标是使得移动机器人能够在未知或部分已知环境下自主地感知周围环境并且规划路径，以达到预定目标。该系统通常由传感器、地图构建、路径规划、运动控制和局部避障等模块组成，以实现机器人的准确定位、路径规划、避障和运动控制等功能。本着“理论结合工程”的原则，在工程中发现问题，解决问题，最终实现园区场景下安全无人配送。这对师生的实际需求拥有重要意义。本文的主要研究内容是面向园区场景的移动机器人多层级导航系统设计与工程实现，目标是实现移动机器人在园区场景下完成自主导航任务。园区有两大特征：宽广但封闭；动态区域多。此目标要求移动机器人根据先验地图信息与任务需求，自主定位并规划路径、移动避障，最终安全到达指定目的地。本文的主要工作概述如下：

1. 以实验室阿克曼底盘为基础，研究了一种多层次的导航方法，并据此从上至下设计出一套多层次移动机器人自主导航系统，该系统根据在导航任务的规划的先后顺序以及与硬件系统的交互逻辑划分成两个子系统层级，分别为高层导航和底层导航。高层导航的任务目标是根据发布的任务目标点，运用RTK 实现自身定位，在事先构建的园区地图上规划出当前位置与终点之间的全局路径，并且将路径信息——一系列路径点下发给底层导航。其中先验地图的构建使用自采点云数据集通过一系列点云处理算法进行后期处理，得到可供导航系统使用的多层地图——点云、栅格、拓扑地图。以 A* 算法和系统相适应，最终完成整体高层导航的任务。

底层导航的任务目标是根据高层导航的规划结果，以激光雷达实时获取周围环境信息，实现自我定位，在点云环境中实现局部规划，通过简单避障逻辑，避开障碍物，到达指定点，逐次循环，直至循迹至高层导航规划路径点的最后一个，整个导航任务完成。此多层次导航系统优点突出，以高层为底层做指引，防止底层导航算法因局部最优解陷入全局场景下的“迷宫陷阱”，并且以多模块方式进行设计，子系统之间的配合紧密，同时可实现系统的后续优化等工作。

2. 在上述系统设计的基础之上，面对园区场景测试过程中面临的多动态障碍物与部分场景狭小导致的底层避障算法不能胜任的困难，从而致使整个系统园区内运行不鲁棒的问题，以 ORCA (Optimal Reciprocal Collision Avoidance) 为算法基础，结合阿克曼型移动机器人模型约束，并结合目前的导航系统框架中基于运动元的局部规划方法，实现了园区场景下多动态场景的自适应避障。经此算法规划出的移动机器人路径在穿越多个移动障碍物区域时，可保持路径最优且系统鲁棒。

摘 要

3. 根据系统的设计方法与思路，结合成本、性能等多方面考虑，搭建了移动机器人导航系统的硬件体系，并且在校园场景下进行多次测试和实验，验证了系统的功能性和鲁棒性。结合系统的运行逻辑，阐述了当前移动机器人导航系统的信息流向和内部工作原理，深刻论证当前系统的优点与在园区场景下的导航能力。

关键词：移动机器人；阿克曼底盘，多层级导航系统；ORCA 避障；建图定位，路径规划，激光雷达

ABSTRACT

Mobile robot navigation system is one of the important research directions in the field of robotics. Its core goal is to enable mobile robots to autonomously perceive the surrounding environment and plan paths in unknown or partially known environments to achieve predetermined goals. The system is usually composed of modules such as sensors, map construction, path planning, motion control, and local obstacle avoidance to realize functions such as accurate positioning, path planning, obstacle avoidance, and motion control of the robot. In line with the principle of "combining theory with engineering", problems are found and solved in the engineering, and finally realize safe unmanned delivery in the park scene. This is of great significance to the actual needs of teachers and students.

The main research content of this paper is the design and engineering implementation of the multi-level navigation system for mobile robots in the park scene. The goal is to realize the autonomous navigation tasks of mobile robots in the park scene. This goal requires the mobile robot to autonomously locate and plan paths, move to avoid obstacles, and finally reach the designated destination safely according to prior map information and task requirements. The main work of this paper is summarized as follows:

1. Based on the Ackerman chassis of the laboratory, a set of multi-level mobile robot autonomous navigation system is designed from top to bottom. The system levels are called high-level navigation and bottom-level navigation respectively. The task goal of high-level navigation is to use RTK to realize self-positioning based on the released task target points, plan the global path between the current location and the end point on the pre-constructed park map, and send the path information—a series of path points. For the bottom-level navigation; the task goal of the bottom-level navigation is to obtain the surrounding environment information in real time with the laser radar according to the planning results of the high-level navigation, realize self-positioning, realize local planning in the point cloud environment, and avoid obstacles through simple obstacle avoidance logic. Arrive at the designated point, cycle successively until the last one of the high-level navigation planning path point is traced, and the entire navigation task is completed. This multi-level navigation system has outstanding advantages. It uses the high-level as the bottom-level guidance to prevent the bottom-level navigation algorithm from falling into the "maze trap" in the global scene due to local optimal solu-

Abstract

tions. It is designed in a multi-module manner, and the cooperation between subsystems is close. At the same time, the follow-up optimization of the system can be realized.

2. On the basis of the above system design, in the face of multi-dynamic obstacles in the test process in the park scene and the narrowness of some scenes, the underlying algorithm planning fell into a "dead end", resulting in the problem of unrobust operation of the entire system park , based on the ORCA (Optimal Reciprocal Collision Avoidance) algorithm, combined with the Ackermann-type mobile robot model constraints, and combined with the current navigation system framework, the adaptive obstacle avoidance and local planning optimization of multi-dynamic scenes in the park scene are realized. untie. The path of the mobile robot planned by this algorithm keeps the path optimal and robust when passing through multiple moving obstacle areas.

3. According to the design method and idea of the system, combined with cost, performance and other considerations, the hardware system of the mobile robot navigation system was built, and multiple tests and experiments were carried out in the campus scene to verify the functionality and robustness of the system sex. Combined with the operating logic of the system, the information flow and internal working principle of the current mobile robot navigation system are expounded, and the advantages of the current system and the specific performance in the park scene are deeply demonstrated.

Key Words: Mobile robot; Ackerman chassis; multi-level navigation system; ORCA obstacle avoidance; mapping positioning; path planning; laser radar

目 录

第 1 章 绪论 ······	1
1.1 研究背景与研究意义 ······	1
1.2 移动机器人导航系统研究现状及其分析 ······	2
1.2.1 移动机器人导航系统的发展历程 ······	3
1.2.2 移动机器人硬件平台研究现状 ······	4
1.2.3 移动机器人导航避障研究现状 ······	8
1.2.4 研究现状分析 ······	9
1.3 研究内容 ······	10
1.4 论文组织结构 ······	12
1.4.1 章节结构图 ······	12
1.5 脚注 ······	13
1.5.1 二级节标题 ······	13
第 2 章 移动机器人导航技术及相关知识 ······	14
2.1 引言 ······	14
2.2 移动机器人导航概述 ······	14
2.3 感知模块技术基础 ······	16
2.3.1 激光雷达 ······	16
2.3.2 摄像头 ······	18
2.4 环境建模 ······	18
2.4.1 点云地图 ······	19
2.4.2 栅格地图 ······	20
2.4.3 拓扑地图 ······	21
2.5 决策规划 ······	21
2.5.1 局部规划 ······	21
2.5.2 全局规划 ······	22
2.6 定位模块 ······	23
2.6.1 高精度车载组合导航定位模块 ······	23
2.6.2 激光里程计和视觉里程计 ······	25
2.7 运动控制 ······	26
2.8 局部避障 ······	27

目 录

第 3 章 多层级导航系统方法研究与算法设计方案	29
3.1 引言	29
3.2 基于全局地图的高层导航方法	30
3.2.1 基于 RTK 的多层地图构建	30
3.2.2 基于距离地图的全局路径规划算法	35
3.3 基于局部点云的底层导航方法研究	36
3.3.1 基于 Fast-GICP 的局部点云	36
3.3.2 局部点云的分割与聚类	40
3.3.3 底层局部规划器设计	44
3.3.4 运动控制	49
3.3.5 局部区域的底层导航方法验证实验	49
第 4 章 基于 ORCA 的移动机器人导航避障系统	57
4.1 引言	57
4.2 ORCA 算法基本原理	57
4.2.1 ORCA 在导航中应用的问题描述	58
4.2.2 ORCA 的求解	61
4.2.3 多个机器人碰撞的情况	62
4.3 基于 ORCA 的系统级避障策略	63
4.3.1 多动态区域的移动物体观测与速度计算	63
4.3.2 ORCA 与系统的结合方案	64
4.4 动态避障系统的仿真实验	66
4.4.1 实验一：4 动态障碍物遭遇实验	68
4.4.2 实验二：8 动态障碍物遭遇实验	69
4.5 动态避障系统的实际场景实验	71
第 5 章 硬件系统、软件算法架构设计与系统信息流向	77
5.1 引言	77
5.2 硬件系统设计	77
5.2.1 硬件选型	77
5.2.2 硬件连接与信息交互	82
5.3 导航系统的实现及其信息流向	83
5.4 底层导航的架构与信息流程	84
第 6 章 总结与展望	85

目 录

第 7 章 数学 ······	86
7.1 数学符号和公式 ······	86
7.2 量和单位 ······	87
7.3 定理和证明 ······	87
参考文献 ······	89
附录 A 补充材料 ······	92
A.1 补充章节 ······	92
致谢 ······	93
在读期间发表的学术论文与取得的研究成果 ······	94

符 号 说 明

- a The number of angels per unit area
 N The number of angels per needle point
 A The area of the needle point
 σ The total mass of angels per unit area
 m The mass of one angel
 $\sum_{i=1}^n a_i$ The sum of a_i

第1章 绪论

1.1 研究背景与研究意义

移动智能机器人是人工智能大背景下的热门研究方向，也是未来国际化竞争的重要方向，随着科技的飞速发展，智能化技术将开启下一个工业革命的大门。谁能在无人智能化技术上领先，谁便在未来的国际话语竞争与产业升级中掌握先机。“中国制造 2025 计划”将高新技术产业作为国家战略之一，而移动智能机器人技术作为高新技术的核心体现之一，自有其璀璨之处。移动智能机器人技术是一门集硬件科学、定位与导航技术、测量与建图科学等相关学科为一体的交叉技术，是一个由传感器、执行器、电源管理以及软件算法组成的复杂控制系统，同时，具备环境感知、路径规划、动态分析、移动避障等多种功能于一体。广义上的移动智能机器人技术包含了无人驾驶、无人配送等多个应用技术方向，除了这些简单的生活用途之外，医疗机器人、战争无人车等特殊用途的机器人也逐渐登上舞台。目前在移动智能机器人技术的应用方向上，走在前沿的技术公司已经有了相当惊艳的产品。一些明星公司例如百度、智行者、酷哇机器人、波士顿等的代表产品如图 1.1 所示。

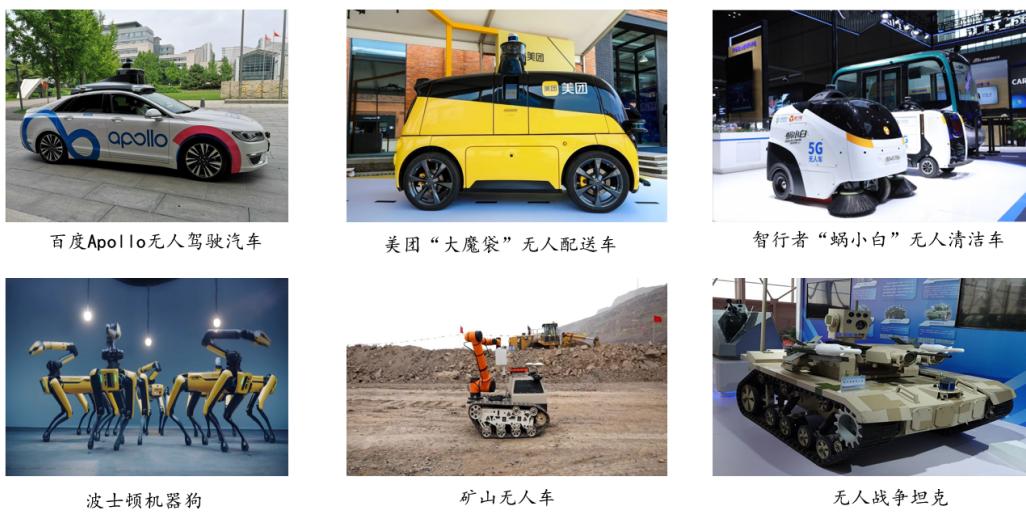


图 1.1 用于各种场景的移动智能机器人

在移动智能机器人领域，移动机器人得到了最广泛的关注与研究。移动机器人的导航技术是其实现目的性运动的最基础功能之一，依托于环境感知、路径规划、定位建图等多个模块的复杂信息，整合信息后将控制信号下发给执行器件，最终完成移动机器人的有目的性运动。可以认为导航系统是移动机器人由单纯的实验室验证模型走向功能性使用的第一步也是最关键的一步。随着劳动力市场的用人成本大幅度提升，使用移动机器人代替工人进行重复、无聊的工作

成了提高生产效率，节约生产成本的重要途径。例如，目前移动机器人在多个园区已经开展了无人配送、无人清扫等多方面的广泛应用，而在这个过程中，移动机器人的导航技术则是这些功能实现的前提，导航技术的最根本任务就是依托指令的目的化运行，导航技术将移动智能机器人从以往的点对点单一指令化工作，提升至目前可在封闭园区内（甚至开放场景下）动态地避开行人等动态障碍物，实施规划、随机应变地完成非固定化路线的导航任务。在移动机器人的最基础功能中，导航系统因其本身在无人设备或系统上的泛化应用，得到了多方机构乃至高校实验室的研究青睐。不仅是在生产过程中，在救火救灾、抢险以及特种任务中，智能无人平台都发挥了巨大的优势。2020年初的突发疫情，更是为移动智能机器人的发展提供了舞台。在国内外多处人流密集场所，封控小区以及其他对非必要性人为接触有需求的场合，智能无人平台凭借其出色的运动性能，获得了多家医院以及酒店的使用青睐。我们有理由相信，在我们国家取得了伟大的疫情防控胜利的成果之中，移动智能机器人的贡献是极大的。目前的移动机器人已经应用在了各行各业之中，伴随着移动机器人的应用，导航系统的发展也与时俱进。以往的导航系统仅仅是简单的对目的地的响应，而随着移动机器人的应用环境愈发复杂：从室内到室外，从静态环境到动态环境，从小范围使用到园区场景下部署，每一次场景的改变，都对移动机器人的导航系统产生了极大的要求，也对导航系统的性能提出了更为严苛的挑战。

为了响应国家“中国制造 2025 计划”的号召，同时因我的研究生生涯都在疫情中度过，深刻认识到了移动机器人技术在园区场景下的广泛运用的意义，本文旨在围绕校园场景下这一园区环境的代表性场所，研究并且设计开发出一套适合本地校园场景等园区场景下的导航系统，满足师生日常对于配送、定点货运以及危险科研任务等实地需求。在此过程中，解决局部最优以及园区场景下动态物体较多等重大问题，是本研究的亮点，同时也是探索移动机器人技术在园区场景下的延伸需求关键突破。

1.2 移动机器人导航系统研究现状及其分析

移动机器人的导航系统是指移动机器人在接收到目的地信息后，对目标点信息处理，根据自身实时定位，在可行驶区域上规划从当前位置到达目标点位置的可行路径，并且沿着既定路径行驶，期间通过激光雷达或者视觉信息实时感知周边环境，对外部环境的改变做出实时反应，安全高效、无碰撞地到达目标点的功能系统。移动机器人的导航系统是无人车类智能化系统的最基础、最关键技术。一般来说，任何无人智能设备的导航系统需要结合多个底层模块的功能，加工处理多种输入输出信息，而后将已有信息作为输入，通过导航系统的计算，输

出相应的系统控制信息给到执行部件，最终完成智能无人设备的导航功能。由于导航系统上连无人设备的信息决策层，下至底盘、激光雷达等硬件层，所以导航系统是一个复杂的耦合系统。本部分会对导航系统的需求进行分析，并阐述导航系统的发展历程，关注在移动机器人导航系统在各个发展阶段的代表性作品，分析各阶段方法的优缺点。最后，对现有导航系统的问题进行总结归纳，并对导航系统最关键的安全性问题上进行着重分析，并给出解决方案。

1.2.1 移动机器人导航系统的发展历程

移动机器人的导航技术是一项综合性交叉技术，主要分为前端感知、建图定位、路径规划、以及决策执行等模块。最早的移动机器人出现于 20 世纪中叶，此时距离汽车的发明已经过去了半个多世纪，随着早期汽车的成熟，智能化汽车的概念应运而生。1956 年，美国通用汽车公司制造出了第一辆无人车实体 Firebird II，并首次提出自动导航的概念。在第一辆移动机器人问世后第十年，美国斯坦福大学的研究所 SRI 人工智能研究中心研发出具有车轮结构的机器人 Shakey，这标志着移动机器人技术从单纯的汽车开始往其他的应用方向蔓延。在今天的视角看来，尽管这个无人车轮机器人内置的传感器和软件系统十分落后，但其开创了自动导航系统的结构先河——软硬件结合的导航系统。此时的无人车导航系统智能化依旧亟待发展，并且导航系统与外界的信息交互十分有限，导航系统的首要目的就是安全，对于汽车来说，最直观的安全就是不发生碰撞，于是在 1977 年，日本的朱波工程研究实验室开发出了第一辆基于摄像头检测前方标记或者其他导航信息的汽车。这辆移动机器人可在轨道的辅助下实现 30km/h 的“飞驰”，同时标志着人们开始从“视觉”角度分析无人车导航系统的感知端。随着 1973 年以后 GPS 系统的展露头角，美国国防高级研究计划局启动了“自主车辆计划”，通过摄像头检测地形，并且由计算系统给出导航路线。到了 21 世纪之后，随着半导体技术以及摄像头器件的飞跃，感知技术逐渐由之前的传统检测方法向着深度学习以及神经网络方向过渡。例如对摄像头采集到的图像数据进行道路识别、车道线检测以及障碍物分割，并且随着技术演进，轻量化的网络也开始登上嵌入式系统的舞台，可以实时检测当前环境中的动态物体，并交给前端感知进行分割，对分割后的静态信息二次处理，保障对动态障碍物的实施避障。比如科学界广泛应用的 YOLO 网络已经迭代到了 YOLO v8，足以见证感知方向的突飞猛进。在视觉方向不断发展的同时，激光雷达的逐渐成熟以及量化成型，为感知融合创造了极大利好。视觉的优点是信息量丰富、易于动态网络检测和可视化，但其因易受极端天气和光照环境的影响，这极大制约了移动机器人的应用场景。而激光雷达可以在极端天气和黑暗状态下正常使用，这无疑是“黑夜 + 白天”极佳组合。目前国内外涌现出了一大批优秀的车载激光雷达供应商，例如美

国的 Velodyne 公司和国内的禾赛、速腾聚创等。

一般来说，园区导航系统的建图定位技术不同于 SLAM 系统中的即时定位建图。建图主要是通过 SLAM 技术或者相关的数据采集、后期拼接优化等方法进行处理，得到可用于高精度定位的静态地图。在导航系统中，地图不是目的，是导航定位的工具。目前的主流定位方法是视觉定位、激光雷达定位以及二者的融合定位方法。例如目前火热的 hdl-global-localization(NDT) 方法，就是根据点云地图的特征信息，结合激光雷达此时采集到的点云信息，使用 scan-to-map 的定位方法，实现移动机器人在先验点云地图中的厘米级定位。目前的定位方法一般都搭配卫星定位来增加定位的精准度。例如使用 GPS 或者 RTK 的定位，其中 RTK 的定位精度可保证在 10cm 以内，对于园区的使用来说，这是一个在可靠范围内的定位精度。

1.2.2 移动机器人硬件平台研究现状

移动机器人一直都是各大机构、高校以及公司的研究热点，在此过程中，各种功能丰富、适用场景多元的移动机器人硬件平台层出不穷，移动机器人的导航系统也随着硬件平台的发展而完善，更加完备的硬件平台也是更高级的导航系统的载体。硬件平台与导航系统是相辅相成且不可分割的，导航系统的设计初衷决定了硬件平台的选型与搭建方式。针对于移动机器人的应用场景类别，总体可分为小场景下的移动机器人，例如室内服务机器人、清洁机器人、工厂机器人等，还有大场景下的移动机器人，例如园区场景、无人驾驶场景等。

1. 室内移动机器人——小规模场景

最早的移动机器人导航系统是从室内移动机器人的研究开始的。一般来说，业界普遍达成共识的是第一台移动机器人是来自于斯坦福研究所人工智能中心的“Shakey”^[1]，如图... (a) 所示，这台机器人“先驱”配备有摄像头和碰撞传感器，其导航系统是基于符号推理和逻辑推断的，通过摄像头对环境的不停感知，进行自主路径规划，从而实现在室内场景的特定导航任务^[2]，例如寻找特定位置等。但当时的计算条件，但是对环境感知分析一项，就需要花费数小时进行，其存在更多是一种象征意义，标志着移动机器人时代的开启。随后，产业化的移动机器人开始出现，1995 年，美国机器人公司 ActivMedia Robotics 推出了一款可二次开发的室内移动机器人 Pioneer^[3]，其导航系统通过激光雷达和摄像头等多种方式进行感知和避障，多用于室内场景的巡逻、运输和建图等。其底盘为差速转轮模型，由于其成本可控，且套件丰富，被广泛用于大学和研究机构的机器人研究中。但是初期的移动机器人因传感器价格昂贵，计算机算法不足等局限性，迟迟无法解决投入实际使用的难点，这极大地制约了机器人的发展。2000 年之后，随着计算机技术的飞跃，移动机器人迎来了发展的黄金期，仅在 10 年以前，

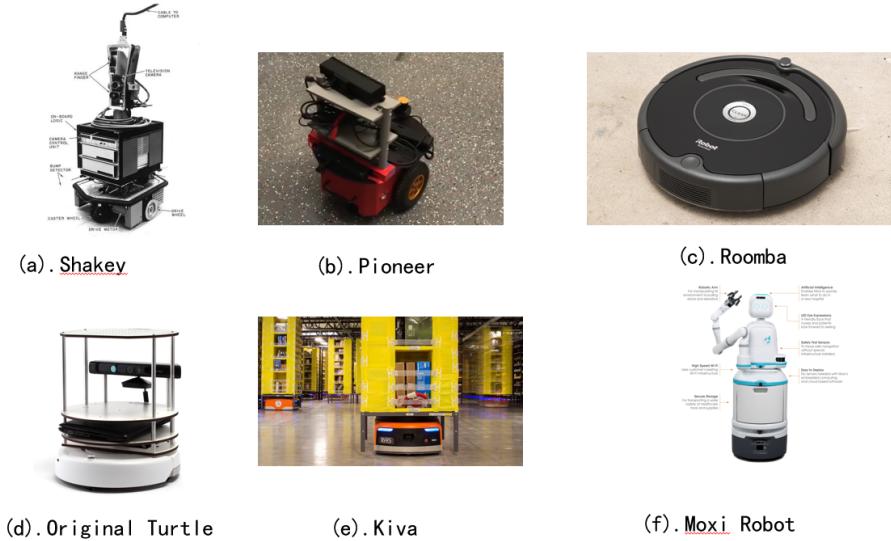


图 1.2 室内移动机器人代表作品

涌现出的优秀代表作品数不胜数，这个时期也是移动机器人逐渐进入商用的关键时期，iRobot 公司于 2002 年推出一款机器人，名为 Roomba^[4]，它可以通过红外线、声音、激光雷达等多种传感器来感知周围环境，实现自主导航和障碍物避让。机器人采用的导航算法是墙隅检测算法（Wall Following Algorithm），它可以通过检测墙面来实现自主运行和避障。由于性能优秀，目前 Roomba 已经迭代了多代，功能也愈发强大。随后伴随着 ROS 系统一并问世的 Turtlebot^[5]给即将到来的机器人时代注入了更大的动力，它采用激光雷达作为感知器件，也可加入深度相机，使用当时并不成熟的 SLAM 技术进行定位，导航系统已经初具现在移动机器人导航系统的模式，可在室内进行多种任务的执行，同时为机器人的研究和发展提供了良好的硬件方案。亚马逊旗下的 Robotics 公司推出了专为物流运输而设计的移动机器人 Kiva^[6]，它的导航系统基于自主引导技术，采用二维条码标识（2D barcodes）的方式实现自主定位和导航。仓库地面上铺设了一些特殊的标记，每个标记都有独特的标识码。机器人通过底部的摄像头识别这些标记，从而确定自己在地图上的位置，进而计算出最优路径，避开障碍物，到达指定位置。由于其工作环境主要是室内仓库，因此采用这种条码式的标记引导导航十分适合。除了电子标签导航方式之外，最新的 Moxi^[7]机器人采用了多种导航技术，包括激光雷达、摄像头、惯性测量单元（IMU）等，以实现高精度的室内定位和导航。具体来说，Moxi 机器人采用激光雷达进行地图构建和建立坐标系，同时结合多个摄像头进行实时环境感知和障碍物检测，IMU 则用于进行里程计计算和姿态估计。Moxi 还使用了机器学习技术，以学习和识别环境中的人和物体，并根据环境变化实时调整行动路径和速度。

2. 室外移动机器人——大规模场景

为室外场景设计移动机器人相比于室内来说，因场景规模较大，难度总体上升一个量级，且室外环境多种多样，园区、矿井、野外等室外环境千变万化，这个过程中也涌现出了一大批优秀的移动机器人设计机构及问世的移动机器人。世界上第一台能够自主导航的车型机器人来自于美国斯坦福大学人工智能实验室的一个早期机器人项目，Stanford Cart^[8]配备有激光雷达、摄像头作为感知传感器，并可以基于当前环境做简单路径规划，躲避动态障碍物。但当时受限于技术的瓶颈，Cart 的导航系统并不能对环境进行有效感知，仅仅是可以识别地上的白线进行跟随控制，但这样已经是耗费了计算系统的大量算力。但 Cart 的问世极大鼓舞了研究者们对移动机器人乃至自动驾驶的探索热情，并奠定了摄像头在移动机器人领域的感知地位。

在室外矿场环境、自然洞穴以及其他复杂环境下，机器人的导航和移动都将变得困难，为攻克这个问题，美国国防高级研究计划局（DARPA）举办了系列比赛——“地下挑战赛”。该赛事目标是开发出能够在地下环境中自主导航、探测、定位、通讯和合作的移动机器人系统，以提高在复杂环境中的人机协作能力。于 2007 年参赛的 GorundHog^[9]是一款由卡内基梅隆大学的机器人研究所（Robotics Institute）开发的移动机器人，配备有 Sick 型号的前后方向的激光雷达以及 360° 全向的激光雷达，具备激光感知的同时还备有一个摄像头用于视觉感知。其导航方式是基于激光雷达的 SLAM 方法激光雷达用于建图和障碍物检测，惯性导航系统则用于估计车辆的位置和姿态。其中导航系统所使用的地图则是执行路径规划和障碍物避免的关键数据。同时 CMU 团队开发的全套自主开源导航系统算法所使用的机器人 CMU-EXPLORATION^[10]是一款多场景导航探索机器人，配备有用于导航与测绘的激光雷达 Velodyne Puck、一个 640×360 分辨率的摄像头和一个基于 MEMS 的 IMU，以及一只专门用于状态估计的激光雷达，如图所示。低矮的车身并搭配两轮差速的运动方式，使该平台能够在狭窄、复杂的空间中快速穿梭，实现在校园、楼层等多种场景下的导航与探索任务^[11-12]。Boston Dynamics 和 ANYbotics 研发的机器狗已经可在全地形条件下执行任务，由于摆脱了轮子的束缚，使得其在任意条件下的移动都可以实现，通常都配有激光雷达、摄像头、IMU 和距离传感器，通过使用深度学习算法和传感器进行导航和避障，两只机器狗的名字分别是 Spot 和 ANYmal^[13]。并且已经有相当多使用 Spot 和 ANYmal 进行导航实验的学术成果，例如使用机器狗导航移动和到特定位置抓取物品^[14]。在国内，移动机器人的发展亦如火如荼，例如美团的“大魔袋”移动机器人主要是在小区和园区等封闭室外场景进行区域内的快递、外卖配送。该机器人采用全轮驱动、4 轮独立悬挂设计，可以适应各种路况，同时具备强大的动态避障能力。其主要硬件包括雷达、摄像头、激光传感器等，以及可拆卸的货

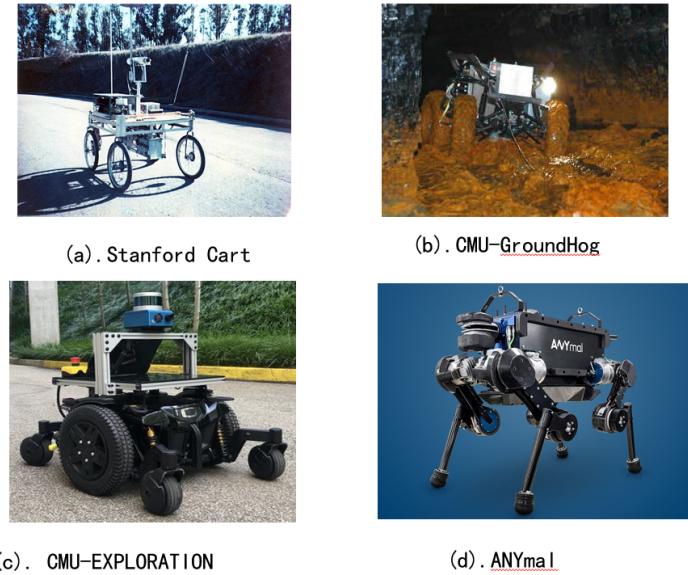


图 1.3 室外移动机器人代表

箱。导航系统采用激光雷达和深度相机实现地图构建、定位和路径规划。首先，激光雷达扫描环境获取地形信息，构建地图。然后，深度相机对环境进行实时感知，包括路面状况、障碍物等，用于实现动态避障和路径规划^[15]。在定位方面，美团大魔袋采用了视觉惯性融合（Visual-Inertial Fusion, VIF）技术，能够实现高精度的自主定位。同时美团大魔袋还具有多种交互方式，如语音、人脸识别等，可以实现人机互动和用户认证等功能。此外，美团大魔袋还支持远程控制和实时监控，保证了配送安全性和效率。

由上述移动机器人的硬件平台研究发展可以看出，移动机器人由室内转向室外并向着多元化使用场景延伸是一个不可阻挡的趋势。计算机技术的大力和发展和传感器技术的不断进步是推动移动机器人硬件导航系统不断演进的基础，同时，更加先进的算法适配上更加优秀的硬件系统，使得移动机器人的性能有了质的飞跃，逐渐从实验室模型 demo 到商业化场景的落地，产生了巨大的实用价值。在硬件系统的进化过程中，导航系统的搭建方式也有了更加科学的规划，主要依靠于激光雷达和摄像头作为感知器件的路线被证明其科学性，辅以 IMU 等惯性测量技术的卫星定位以及 SLAM 技术是目前导航系统的主要定位手段，且随着定位技术的成熟化，必将推动导航系统中路径规划和避障技术的持续进步。本文的研究更加关注于当前各大高校、研究机构以及商业公司推出的移动机器人导航系统的架构搭建以及其中涉及到的传感器技术。根据规划，目前阶段使用激光雷达作为在园区场景内导航系统的主要感知传感器是十分科学，因此本文中的系统构建选择 Velodyne VLP-16 激光雷达作为导航系统的主要感知器件。

1.2.3 移动机器人导航避障研究现状

1. 静态障碍物避碰

基于代价地图的方法：这种方法使用代价地图来表示机器人所处的环境，其中每个网格单元格都包含一个代价值，表示该区域对机器人行动的限制程度。代价地图可以通过各种感知设备获得，例如激光雷达、摄像头、超声波传感器等。机器人可以通过在代价地图上进行路径规划来避免碰撞。

基于几何形状的方法：这种方法利用障碍物的几何形状信息，例如大小、形状和位置，来计算机器人的运动轨迹，以避免碰撞。常用的基于几何形状的方法包括多边形轮廓算法和边界表示法等。

基于局部传感器的方法：这种方法利用机器人附近的传感器信息，例如超声波传感器、激光雷达等，检测机器人周围的障碍物。机器人可以通过实时分析传感器数据来规划避障路径，以避免碰撞。

基于网格法的方法：这种方法将环境划分成一个网格，然后根据障碍物的位置将一些网格标记为不可通过，再使用搜索算法进行路径规划。常用的基于网格法的方法包括 D 算法和 A 算法等。

2. 动态障碍物避碰

导航系统中在园区场景下的动态障碍物碰撞避免方法十分多样，有基于速度障碍（Velocity Obstacle）的方法，基于强化学习的方法，基于模型预测控制（Model Predictive Control）的方法和基于人工势场的方法。

速度障碍的概念被引入移动机器人导航避障领域是 2008 年时，van den Berg et al.^[16]提出了一种名为”reciprocal velocity obstacle”（RVO）的方法，用于实现多智能体系统中的实时导航和避障。在移动机器人多智能体避障领域，传统方法是通过预测其他智能体的行动来避免碰撞，但这种方法需要对其他智能体的意图进行预测，而这通常很难。RVO 方法不需要对其他智能体的意图运动进行预测，而是通过计算每个智能体之间的速度障碍来避免碰撞。RVO 的方法在复杂场景中容易出现局部最优解的问题，于是 ORCA^[17]的方案问世，这一方法由 RVO 的提出团队在原 RVO 方法的基础上改进得来，称为 “Optimal Reciprocal Collision Avoidance”。与 RVO 不同，ORCA 在计算方式上使用线性优化技术来计算最优速度，以便避免碰撞和尽可能地接近目标。同时考虑了其他智能体的速度和位置，通过避免与其他智能体发生碰撞来实现全局最优。ORCA 算法的优点在于它能够保证全局最优解，但它的计算复杂度更高。随后，Guy et al.^[18]提出了一种基于 GPU 加速的多智能体避障算法——ClearPath。该算法使用 RVO 算法来计算速度，并利用 GPU 的并行计算能力来加速碰撞检测。实验结果表明，ClearPath 算法在性能上比传统的 CPU 算法要快得多。Hennes et al.^[19]提出了一种基于速

度障碍范式的多机器人碰撞避免系统。通过在每个智能体上使用自适应蒙特卡罗定位来减轻对每一个障碍物精确感知的要求。因现实的实现不可能做到对周边动态的完美感知，需要进一步扩展来处理不准确的定位和消息传递延迟。此算法限制了由定位引入的误差，并将无碰撞运动的计算与定位不确定性相结合。这篇文章开始从应用层面考虑 ORCA 的实用性，为推动 ORCA 在移动机器人导航避障的实地应用做出巨大贡献。在实际应用方面，Snape et al.^[20]提出了一种用于多个独立机器人在差动驱动约束下进行平滑和无碰撞导航的方法。该算法基于最优互惠碰撞避免公式，并保证了机器人轨迹的平滑性和局部无碰撞路径。此方法在运用中结合了差速转轮模型，将 ORCA 在差速转轮模型约束下进行计算。在其之后，使用此研究结论进行实地实验的结果均证明了算法的有效性。在此基础之上，Alonso-Mora et al.^[21]提出了一种自行车模型的互相碰撞避免算法（B-ORCA），基于移动机器人的 ORCA 的概念，但进一步保证了在类似汽车的运动约束下的无碰撞运动。B-ORCA 算法的基本原理更广泛地适用于其他运动模型，因为它将速度障碍与通用跟踪控制相结合。通过多个类似汽车的机器人之间的几个仿真实验验证了碰撞避免方面的理论结果。Snape et al.^[22]描述并评估了两种用于多个独立差动驱动机器人的平滑和无碰撞导航的算法。文章基于速度障碍和加速度-速度障碍扩展了互惠碰撞避免算法。并在多个差速机器人上进行了实验，实验证明了文章所提的方法的有效性和实用性。

1.2.4 研究现状分析

目前移动机器人的导航系统的在架构和硬件体系方面已经相当成熟，不同的系统架构可以支持移动机器人在不同的场景下开展导航任务，但对于不同的任务场景之间，移动机器人的导航系统并不能依照同一范式进行支撑或者构建。因此，这给移动机器人在不同场景下的应用以及技术支持带来了巨大的困难。在小规模场景下的工作的移动机器人系统很难通过一定的模块更改适用在大规模的场景之下，大规模场景下的移动机器人导航系统又不像小场景下的移动机器人导航一般只考虑局部规划即可，还需要进行全局范围内的规划。因此我们期望有一种通用的移动机器人导航体系架构，在这个体系架构下，无论是小规模场景的移动机器人导航系统还是应用于园区的导航系统，均可以通过系统的模块更改做相应的算法适配。遗憾的是，目前的开源机器人系统完全依赖于对特定环境的适应，而不将移动机器人的导航系统统一到某个架构下，所以我们的工作虽然是针对于园区场景展开，但是注重了模块之间的独立性和二次开发。经过简单的更改便可以将系统应用于小规模场景。

同时在园区内应用时，关于移动机器人在园区场景下的穿越多动态区域时规划混乱问题，本文发现关于 ORCA 的讨论一直处于理论和实际应用的边界之

中，即使当前的研究已经深入到 ORCA 和深度学习等人工智能方法结合，但仍然在实际应用中存在巨大间隙，未解决 ORCA 与阿克曼模型之间存在的约束冲突问题。同时目前的研究在进行 ORCA 的计算之时，没有实地导航系统的研究佐证，本文期望在导航系统的局部规划中有机嵌入 ORCA 避障模块，使得园区的移动机器人在进入多动态区域时，以导航系统自带的基于运动元的规划避障方式为主，为 ORCA 提供当前导航系统下的最优参考速度，基于此最优参考速度，开展 ORCA 的线性计算，并且反馈得到的计算结果用以与导航框架和阿克曼模型进行二次约束处理，最终得到避障规划结果。同时由于 ORCA 的计算消耗是巨大的，且目前的研究都是基于已知目前移动机器人周围所有动态物体的前提下进行的，不能有效解决突发情况，例如园区内随时可能出现的突入场景的动态障碍物，因此本文将动态点云的避障同时引入 ORCA 的方法之中，以导航系统的本身的避障为 ORCA 算法提供突发动态障碍物的解决方案，并为 ORCA 算法的最优速度提供参考初值；而 ORCA 算法则去掉对静态障碍物的计算，负责在阿克曼模型约束之下计算穿越多动态区域的最佳速度。

Give a description of the above

1.3 研究内容

据上所论，本文的研究主容主要是基于移动机器人的导航系统研究现状——不同场景下设计的移动机器人之间很难互用，移动机器人导航系统目前存在的多种多样的设计方案，且设计方案之间难以协调沟通，同一设计方案中各模块协调性不足等问题，提出一种多层次级设计思路的导航方法，此方法由园区移动机器人的导航问题为切入点，将全局性的导航和局部性的导航子系统分模块设计，并且各系统内部模块之间独立工作，可以任意对定位、避障等模块进行二次开发。因为导航系统具备有适应大小规模场景的导航通用架构，易于进行方案的移植。在导航方法的研究与设计过程中，结合了多种性能优越的类别算法，例如底层地形分析与点云聚类算法以及高层中全局地图构建过程的栅格占据状态贝叶斯估计法则与全局路径规划的 A* 算法。此系统模式简明，性能优越，在园区场景下的测试过程中表现良好。无论是全局的路径规划任务还是小规模的室内规划任务均可出色完成。系统的亮点在于不依靠单一的路径规划算法解决全局状态下的导航问题，使用“多段规划”思想将全局状态下的导航任务以高层和底层两个导航子系统做切分，高层将全局规划任务划分为各个小规模区域内的局部规划任务，再由底层导航以局部规划形式完成小规模区域内的导航。关于导航系统的研究主要分为两个子导航系统的方法：

1. 高层导航方法：以 Velodyne VLP-16 激光雷达为传感器，采集园区场景下

的数据集，并提取数据集中的可行驶区域，并去除其中的动态物体信息，保留全局场景中存在的静态可行驶区域部分，将可行驶区域以点云地图、栅格地图和拓扑地图的形式存储起来，作为离线的先验地图。高层导航中的路径规划模块读取先验地图，当得到任务目标的终点位置，通过全局定位系统确定自身位置，启动全局场景下的规划，规划在静态的先验地图上进行，且规划只针对于可行驶区域，因此规划出的路径不受实时情况的干扰，只考虑路径的代价。随后规划模块会根据自身定位与目标点位置规划出一条全局路径 $PATH$ ，保留为一系列路径点，设任意一个路径点为 P_i ，则 $P_i \in PATH$ ，路径点随后会按照车身所处位置，由近及远逐渐下发给底层导航，直至 $PATH$ 中最后一点下发完毕。

2. 底层导航方法：底层导航的核心是局部规划器模块，在接收到高层导航下发的路径点之后，立即开始工作。首先底层导航的局部点云构建模块会构建出移动机器人周围一定范围内的点云环境，然后根据局部点位模块，底层导航会掌握此刻移动机器人在环境中的位姿，以及局部环境中 P_i 的位置。移动机器人为通用阿克曼底盘模型，实现离线构建的以后轮中心为 `base_link` 的运动元路径组，所有的运动元满足阿克曼模型约束，根据自身与目标点 P_i 的朝向角度差距，选择其中的某一条路径，选择算法基于朝向角度偏差、障碍物距离等权重后综合分析。选择出路径的情况下，根据构建的运动元选择对应的转向角度与速度，运动控制模块开始工作，通过 CAN 盒驱动程序将运动速度与转向角度解算为底盘可识别的数字驱动信号，数字驱动信号会驱动电机并控制转向轮，完成行为的表达。循环往复，当 $PATH$ 中最后一个路径点的抵达任务完成之后，整体系统结束运行。

以上便是两个子系统的导航方法与设计思路。此系统已具备在静态环境以及极少动态障碍物的情况进行导航的能力。但当系统运行在园区多动态场景下时，因路底层规导航采用的路径规划模块和避障模块耦合度较高，此时多动态的移动性将会使系统陷入规划混乱和避障提前裕量不足的困境，甚至可能会因此陷入局部最后的死局。

针对园区场景下穿越多动态障碍物区域时出现的避障不鲁棒或者局部规划混乱的问题，基于现有导航框架，提出了一种基于 ORCA 与运动元规划法相结合的动态导航避障方法。此方法的规划基于底层导航的规划路径与移动方向，将此方向设置为 ORCA 方法的最优速度方向，动态避障模块根据周围目前的动态障碍物的速度，规划出移动机器人碰撞避免的速度集合，在此速度集合中根据最优速度选择出备用速度。此速度集合由 ORCA 库的默认圆模型计算而来，需要对速度集合结合阿克曼模型进行二次筛选。若备用速度不满足阿克曼模型约束，则取备用速度的方向和阿克曼模型约束的最大角度进行方向矫正，若备用速度方向正确，若备用速度方向约束满足，则按照模型约束选择运动元中最接近备用

速度的一条路径方向，然后辅以 ORCA 库计算出的速度大小。通过对周围动态物体运动信息的不停感知分析，其园区内动态物体为行人和车辆的前提下，速度方向不会突变且有一定延续性，此动态障碍物碰撞避免模块可以最小路径代价规划出穿过多动态障碍物区域的局部路径。此局部路径同时满足了阿克曼模型约束下的角度约束与 ORCA 算法下的最小代价，可以使底层规划模块顺利穿越多动态障碍物区域到达目标点 P_i 。

1.4 论文组织结构

论文的组织架构以建立一种通用的导航系统方法框架为指导思想，不按照功能模块，例如定位、建图或者路径规划等来划分，而按照功能层级划分为底层和高层两部分。第三章详细介绍了高低层导航方法中所涉及到的各种算法细节以及方法等，第四章针对第三章中导航方法存在的无法穿过多动态区域的问题，创新提出一种多动态障碍物的导航避障方法，此避障方法作为功能模块仅在检测到多动态障碍物时启动，离开多动态障碍物区域后则停止运行，因此既是方法又是单独的创新设计，故单列为一章。第五章内容是在前面两章方法设计的基础之上，搭建移动机器人的硬件系统，并将此系统之间的信息传递分别以硬件系统的数据流向和软件系统的消息流向方式进行展现，并且附上相应的实地实验。

论文的架构图可如下表示：

1.4.1 章节结构图



图 1.4 模板图片

1.5 脚注

1.5.1 二级节标题

1. 三级节标题

(1) 四级节标题

① 五级节标题

 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
 incididunt ut labore et dolore magna aliqua. ①

① Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

第2章 移动机器人导航技术及相关知识

2.1 引言

当今，移动机器人已经广泛应用于自动化物流、仓储管理、医院和办公室等领域，使得人们的工作更加高效和便利。导航系统是移动机器人的关键功能模块，移动机器人导航技术是指通过多种传感器获取环境信息，对机器人的位置、速度、方向等状态进行估计和控制，以实现机器人自主移动的过程。其输入信息包括机器人当前位置、目标位置、环境地图、障碍物等，输出信息包括机器人运动的速度、方向、路径等。移动机器人导航系统是一个复杂的耦合系统，同时移动机器人导航技术也是一个多学科的交叉技术。本章节主要探究移动机器人导航系统中所涉及到的关键技术，将组成导航系统的各个关键模块之间的关系和为导航系统提供的功能详细解读以及展示这些关键技术在目前产品以及 demo 模型中的应用。由于导航技术牵扯较多，故本章以导航中各模块涉及到的技术领域为章节，详细展开。

2.2 移动机器人导航概述

移动机器人导航系统是一个复杂的多元系统，主要是通过传感器，例如激光雷达感知环境，同时对自身位置进行定位，在构建的全局先验地图基础上进行路径规划，期间局部规划并且避障，将控制信号交给执行部件，完成移动过程。其中涉及到多个学科门类，在最基础的导航任务中，所需要的相关知识主要有以下几个方面：

1. 传感器技术移动机器人的导航需要依靠传感器技术，例如激光雷达、摄像头、超声波传感器等。这些传感器可以帮助机器人感知周围环境，识别障碍物和地标，并提供机器人的位置和方向信息。
2. 地图建模在移动机器人导航系统中，地图建模是非常重要的环节。地图建模的主要任务是将机器人所在的环境转换成机器人可识别的数字化地图。数字化地图通常包括地图边界、地形高度、障碍物位置和机器人目标位置等信息。常用的地图建模方法包括激光雷达扫描、视觉 SLAM（Simultaneous Localization and Mapping）等。
3. 定位模块，在地图建模的基础上，移动机器人需要掌握自身处于环境中的位姿，定位的精准程度直接影响到移动机器人规划和执行的效果。定位的信息包括机器人目前处于建模后环境的位置以及坐标系位姿等。最常见的定位方法有 SLAM 方法、卫星定位系统。

3. 路径规划在数字化地图上，机器人需要能够规划出一条合适的路径以达到目的地。路径规划的方法通常是基于机器人的位置和目标位置，利用搜索算法、最短路径算法等方法来生成一条合适的路径。其中，常见的路径规划算法包括 A* 算法、Dijkstra 算法、RRT（Rapidly-exploring Random Tree）算法等。

4. 运动控制机器人在移动时需要控制轮子或足部运动，以使其沿着规划好的路径行进。运动控制的目标是实现机器人的准确定位、平稳移动和灵活转向。运动控制方法包括 PID 控制、模糊控制等。

5. 局部避障在移动过程中，机器人需要避开障碍物，这就需要局部避障算法。常见的局部避障算法包括基于代价地图的避障方法、局部避障算法等。

6. 实现和测试最后，将以上技术组合起来，实现一个完整的移动机器人导航系统，并进行测试和优化。测试的方法包括模拟测试和实际场地测试。在模拟测试中，可以使用虚拟仿真软件模拟机器人在不同环境下的导航情况。而在实际场地测试中，需要在实际环境中测试机器人的导航性能，如准确定位、路径规划、避障等方面的表现。测试结果可以用来优化算法和调整系统参数，以提高机器人导航的准确性和效率。总之，移动机器人导航系统的实现需要从传感器技术、地图建模、路径规划、运动控制、局部避障等多个方面进行综合考虑和设计，才能实现一个稳定、可靠、高效的自主导航系统。

其任务示意图可以表示如下：

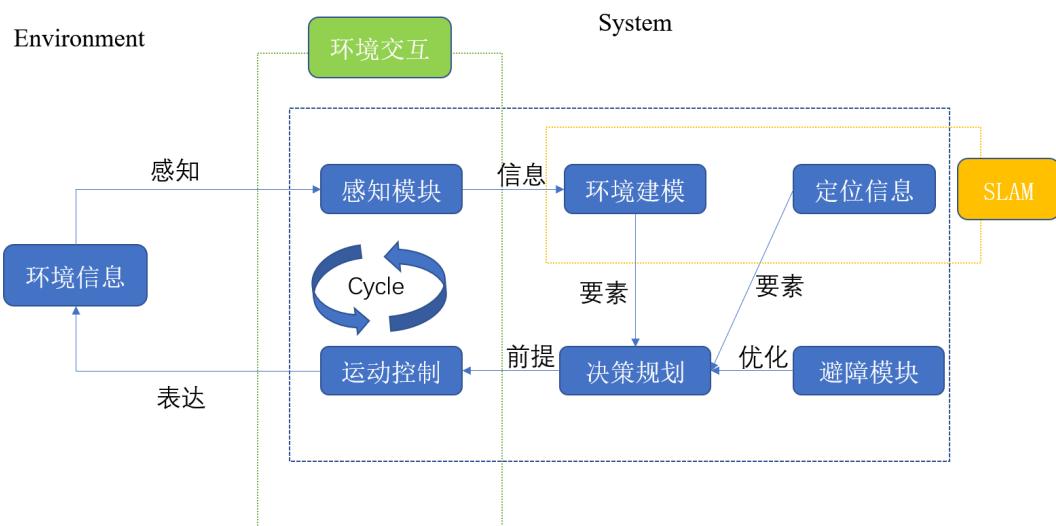


图 2.1 导航任务示意图

图中，感知模块和运动控制是与环境进行交互的模块，在交互过程中不断循环，通过感知模块摄取环境信息，并通过运动控制模块执行表达，体现为在环境中的行为；在导航系统内部，通过感知模块感知到的信息，对环境进行建模，为决策规划提供要素，没有环境信息的导航系统无法执行有效决策规划；决策规划同时需要定位信息作为要素一部分，导航系统了解自身在建模的环境中的位置。

之后便具有决策规划的能力；但导航系统的前提是安全，那么需要避障模块对决策规划结果进行优化，得到无碰撞的规划信息，此信息最终转换成运动控制信号，由具体执行部件，例如电机、车轮等执行，表达为在环境中的行为。

其中环境建模与定位信息可以通过独立模块运行工作，也可以通过 SLAM (Simultaneous Localization and Mapping) 技术提供实时的定位建图。但在园区的全局地图下，为保证系统稳定，多采用单独建图、后期处理的方式取得质量较高的地图；并通过 GPS 全球定位系统等方式进行定位或者辅助定位，具体取决于精度要求。

2.3 感知模块技术基础

感知模块的基础是各种各样的传感器，目前广泛应用于移动机器人行业的传感器主要两种：激光雷达和相机。

2.3.1 激光雷达

激光雷达 (LIDAR) 是一种主动式传感器，通过向周围环境发射激光束，并通过接收反射回来的激光束来获取环境的深度和距离信息，能够提供高精度、高分辨率的环境信息。激光雷达的内部组成主要有以下器件：

1. 激光器：激光雷达使用的激光器需要具有高功率和短脉冲宽度，以实现高精度的测量。一般采用半导体激光器或固体激光器。半导体激光器：半导体激光器是一种基于半导体材料的激光器，通常采用 GaN、InGaN 等 III-V 族化合物半导体材料，具有小体积、低功耗、长寿命、易于集成等优点。半导体激光器的输出波长范围一般在 400-1700nm 之间，适用于大部分激光雷达应用。固体激光器：固体激光器是一种基于晶体或玻璃材料的激光器，通常采用 Nd:YAG、Er:YAG 等材料，具有高功率、高效率、高稳定性等优点。固体激光器的输出波长范围一般在 1000-1500nm 之间，适用于长距离和低散射噪声的应用。

2. 接收器：接收器是用于接收反射回来的激光束的组件。一般采用光电二极管 (photodiode) 或光电探测器 (photodetector)。光电二极管：光电二极管是一种将光转化为电信号的半导体器件，具有响应速度快、灵敏度高、体积小、成本低等优点。光电二极管的工作原理是当光照射到 P-N 结上时，会产生电子-空穴对，从而产生电流信号。光电探测器：光电探测器是一种将光转化为电信号的器件，包括光电二极管、光电倍增管、光电导管等。光电探测器的响应速度和灵敏度较高，适用于激光雷达等高精度应用。

3. 光学器件：激光雷达使用的光学器件包括镜头、光栅和棱镜等。它们的作用是对激光束进行聚焦、分束、旋转等操作。镜头：镜头是一种用于聚焦光线的

光学器件，通常用于激光雷达中的发射器和接收器。激光雷达发射器需要将激光束聚焦成尽可能小的直径，以便实现高精度的距离测量。接收器则需要将反射回来的光线聚焦在光电探测器上，以提高接收效率和精度。

4. 旋转平台：旋转平台是激光雷达中常见的机械部件，可以使激光雷达以一定的速度和角度旋转，以获取360度的空间信息。旋转平台通常采用步进电机或直流电机驱动，可实现高精度的旋转控制和稳定性。

5. 距离测量算法：激光雷达通过测量激光束的往返时间和波长，可以计算出目标物体到激光雷达的距离。常见的距离测量算法包括TOF（Time of Flight）和FMCW（Frequency Modulated Continuous Wave）。TOF算法：即飞行时间测量（Time of Flight）算法，是一种基于激光脉冲往返时间的距离测量算法。其基本原理是通过测量激光束从激光雷达发射出去后反射回来所需的时间来计算物体与激光雷达之间的距离。当激光束从激光雷达发射出去时，它会在空气中以光速c传播。当激光束遇到一个物体时，它的能量将部分地被吸收并转换为热能，同时一部分激光能量将反射回激光雷达。反射激光的时间和激光束的传播速度是已知的，因此可以通过这些信息计算物体的距离。如下图所示，当激光束射向目标物体时，激光雷达记录下发射时间t₁和反射时间t₂。物体到激光雷达的距离R

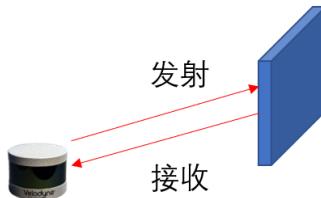


图2.2 基于ToF原理的激光雷达测距示意图

可以通过下式计算得到：

$$L = c(t_2 - t_1)/2 \quad (2.1)$$

其中，c是光速，t₁和t₂是激光束发射和反射的时间差。TOF算法的优点是测量精度高、测量速度快、简单、成本低，但其测量精度受到环境光、物体表面反射率等因素的影响。

FMCW算法：即调频连续波（Frequency Modulated Continuous Wave）算法，是一种利用激光频率变化来测量距离和速度的方法。在FMCW激光雷达中，激光束通过一个运动的镜片反射，产生频率偏移。当激光束与物体相交时，反射激

光会随着物体的运动而产生频率偏移。FMCW 激光雷达通过改变激光的频率来实现距离和速度的测量。激光雷达发射一个连续的线性调频激光波，即下行斜坡。当激光束照射到物体上时，部分激光会反射回来，这部分反射激光的频率会与下行斜坡的频率有所不同。接着，激光雷达发射上行斜坡，这时反射激光会与上行斜坡产生干涉，形成一个频率等于反射激光频率与上行斜坡频率之差的混频信号。接下来，通过对混频信号进行 FFT 变换，可以得到频谱图。频谱图中的峰值对应着物体反射激光的频率，因此可以通过这些信息来计算物体与激光雷达之间的距离。FMCW 算法的优点是精度高、抗干扰能力强，可以同时测量距离和速度，但成本较高。

2.3.2 摄像头

摄像头是一种基于视觉感知的设备，它可以将物体的图像转换成数字信号进行处理。与激光雷达不同，摄像头通过感知光的反射来获取环境中的信息。摄像头通常由透镜、图像传感器和信号处理器等组成。

1. 透镜：摄像头的透镜通常由多个光学镜片组成，其作用是将环境中的光线聚焦到图像传感器上。

2. 图像传感器：通常有两种类型：CCD (Charged Coupled Device) 和 CMOS (Complementary Metal-Oxide-Semiconductor)。CCD 传感器是传统的图像传感器，通过在晶体管阵列中收集和移动电荷来捕获图像。而 CMOS 传感器则是现代摄像头常用的传感器类型，它将光电转换器集成在每个像素中，可以通过集成其他功能来实现更高级的图像处理和计算任务。近年来，CMOS 传感器因其成本低、功耗低和性能高而逐渐取代了 CCD 传感器，成为现代摄像头的主流传感器类型。

3. 信号处理器：将得到的电子信息转换为数字信号，以形成数字图像或视频，通常由多个阶段组成，包括预处理、增益控制、降噪和色彩校正等。这些步骤旨在提高图像的质量和清晰度，并消除图像中的噪声和失真。

摄像头的性能主要由分辨率、帧率、动态范围、色彩深度和信噪比等指标来衡量。分辨率指图像中的像素数量，通常用宽 \times 高来表示。帧率是指摄像头每秒输出的图像数量，通常用赫兹来表示。动态范围是指摄像头能够捕捉到的最大和最小光强之间的范围。色彩深度是指每个像素可以表示的颜色数。信噪比是指信号和噪声之间的比例，它越高则图像的质量越好。

2.4 环境建模

在移动机器人导航领域中，机器人需要一个可供内部系统识别的环境模型，此模型需要能够表达环境的特征信息，例如位置距离和物体结构等。环境建模的

方式有很多种，例如：点云地图（Point Cloud Map）、栅格地图（Grid Map）、拓扑地图（Topological Map）

三维网格地图（3D Grid Map）：将环境划分为三维网格，将每个网格的状态（如障碍物、自由空间等）作为该网格的属性进行存储。与栅格地图类似，但能够更加准确地表示环境中的物体形状和位置。深度图（Depth Map）：使用深度传感器（如激光雷达、RGB-D相机等）获取环境中物体的深度信息，并将其表示为灰度图或RGB图像。适用于需要进行三维重建或深度感知的任务。语义地图（Semantic Map）：将环境中的物体按照其语义类别进行分类和建模，并将其位置、大小、形状等信息保存为语义地图。适用于需要进行高级任务规划、交互和理解的任务。光流场（Optical Flow Field）：使用光流算法从视频流中提取物体在图像中的运动信息，并将其表示为光流场。适用于需要进行移动物体追踪、目标检测和位姿估计的任务。轨迹（Trajectory）：记录机器人或其他物体的运动轨迹，并将其表示为一系列位置坐标和时间戳的数据。适用于需要进行轨迹分析、路径规划和行为分析的任务。场景图（Scene Graph）：将环境中的物体表示为图的节点，将它们之间的关系（如包含、连接、属性等）表示为图的边。适用于需要进行场景理解、关系推理和知识表示的任务。移动机器人领域最常使用的环境建模方式，主要有点云地图、栅格地图和拓扑地图。

2.4.1 点云地图

点云地图将环境中的物体和障碍物的位置、形状等信息保存为点云数据，是一种基于几何形状的建模方式。点云地图适用于需要高精度的环境建模和精确的感知任务，比如自动驾驶、机器人导航等。

在SLAM技术未有突破前，卫星定位系统和激光雷达等传感器采集数据后期建图的方式是主流，此种方式耗时较高，但是建图精度高，可以后期人工纠偏，如今许多园区的高精度地图仍然使用此类方法。

目前更多的在非园区场景下使用基于SLAM方法的同步建图方式，该方式建图速度快，成本低廉，例如Zhang et al.^[23]提出了一文提出了一种实时的激光雷达点云地图构建算法。该算法通过在点云中提取特征点，并基于特征点之间的关系进行位姿估计和运动估计，实现了高效的点云地图构建。在其改进版本LeGO-LOAM中，Shan et al.^[24]将点云数据分为地面点和非地面点两类，并在地面点的处理中引入了高斯混合模型（Gaussian Mixture Model, GMM）对地面点进行建模。通过对地面点进行建模，可以减少环境中非地面点对定位和地图构建的干扰，进一步提高算法的精度和鲁棒性。Wan et al.^[25]涉及到了点云地图构建的问题。该方法通过多传感器融合，将激光雷达数据和摄像头数据进行融合，生成高精度的三维地图，从而实现车辆在复杂城市环境中的高精度定位和导航。如图

是 LOAM 所建的点云地图：

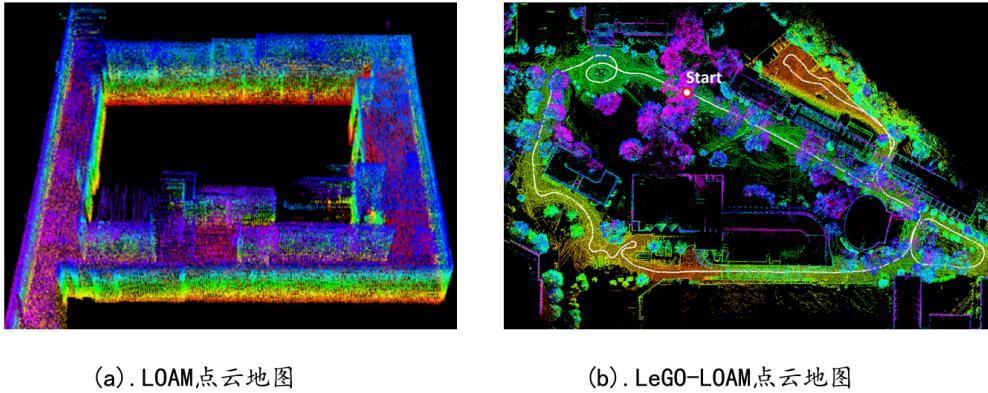


图 2.3 LOAM 簇算法所建点云地图

2.4.2 栅格地图

栅格地图将环境划分为网格，将每个网格的状态（如障碍物、自由空间等）作为该网格的属性进行存储，是一种基于空间状态的建模方式。栅格地图适用于需要快速生成环境模型和实时感知任务，比如移动机器人、无人机等。栅格地图中的占据状态通常由二值化的栅格单元格表示，其中栅格地图的内部占用概率投票方式^[26]为：

$$P(m_i | z_{1:t}, x_{1:t}) = \frac{1}{1 + e^{-l_i}} \quad (2.2)$$

其中 m_i 表示第 i 个栅格单元格的占据状态， $z_{1:t}$ 表示前 t 个时刻的所有观测值， $x_{1:t}$ 表示前 t 个时刻机器人的所有位姿， l_i 表示单元格 i 的 log-odds 值，其计算方式为：

$$l_i = \log \frac{P(m_i = 1 | z_{1:t}, x_{1:t})}{P(m_i = 0 | z_{1:t}, x_{1:t})} \quad (2.3)$$

在实际应用中，栅格地图的占据状态投票概率可以使用不同的方法来计算和更新，例如最大似然法、贝叶斯法、递归贝叶斯法等。最终目的就是对栅格单元格进行占据状态的评估，以进行后续的在栅格地图上的路径规划和避障等任务。移动机器人在栅格地图中的位置具体表现由当前所使用的规划算法决定，例如在 A* 算法^[27]的规划任务中，栅格地图上移动机器人的位置以向量形式表示，例如 (x, y, z) ，但若使用考虑机器人运动学模型约束的算法 Hybrid A*^[28]，则拥有机器人运动学模型约束的移动机器人位姿通常以 (x, y, θ) 表示，其中 θ 表示当前机器人的位置朝向。

2.4.3 拓扑地图

拓扑将环境中的物体和障碍物抽象为节点，将它们之间的关系（如连接关系、距离等）作为图的属性进行存储，是一种基于拓扑关系的建模方式。拓扑地图适用于需要高层次的环境表示和规划任务，比如园区与城市地图上的任务规划等。拓扑地图的构建过程主要是提取地标、建立节点、建立拓扑关系。提取地标的过程就是筛选节点的过程，而后建立相应的节点（node），在节点之间，根据两节点之间的可通行性，判断两节点之间是否有边（edge），若有边，则互相联通。这也称为建立拓扑关系的过程，建立拓扑关系：根据节点之间的空间关系建立拓扑关系。这些关系可以是直接测量的距离或角度，也可以是通过机器学习方法推断得出的。建立拓扑关系的方法有很多种，其中一种常用的方法是使用基于距离的方法，如 Voronoi 图。基于距离的方法将空间分成不同的区域，每个区域都对应一个节点。然后，使用距离信息将这些节点相连，建立拓扑关系。另一种方法是基于相邻性的方法，如最近邻算法或 RANSAC 算法。拓扑地图相对于栅格地图和点云地图相比，1. 可扩展性强：拓扑地图中的节点和边缘可以随时添加或删除，而不需要对整个地图进行重新构建。因此，拓扑地图能够更好地应对环境的变化。2. 可解释性强：拓扑地图中的节点和边缘通常具有语义信息，能够描述环境中的重要位置和关系。因此，拓扑地图能够提供对环境更深入的理解。3. 更高的效率：由于拓扑地图通常是稀疏的，路径规划和定位等算法可以更快速地执行。4. 更少的数据存储：与其他地图形式相比，拓扑地图通常需要存储的数据更少，因为它们只包含有用的拓扑信息。5. 鲁棒性强：拓扑地图中的节点和边缘可以使用不同的噪声滤波器进行平滑和处理，从而提高了定位和规划的鲁棒性。因此，拓扑地图在大型的高层次环境表达和路径规划领域，具有相当广泛的应用，同时，RRT^[29]、RRT*^[30]以及其它衍生规划算法与拓扑地图具有极高的适配性。

2.5 决策规划

决策规划部分可以根据实际应用中所属的层次分为局部规划和全局规划方法。

2.5.1 局部规划

人工势场法：人工势场法是一种基于势场的局部路径规划方法，它通过在机器人周围建立势场，使得机器人沿着势场梯度的方向移动。人工势场法的优点是简单易实现，实时性好，但容易出现局部最小值陷阱问题。人工势场法的基本思想是将机器人和障碍物看作是带电荷的物体，机器人之间相互排斥，机器人与障

碍物之间相互吸引。机器人在势场中的移动方向是势场梯度的方向，即向势能下降最快的方向移动。

滑动窗口法：滑动窗口法是一种基于轨迹的局部路径规划方法，它将机器人的轨迹分为若干个滑动窗口，然后在每个滑动窗口内进行规划。滑动窗口法的优点是可以在不同的窗口内采用不同的路径规划算法，能够更好地适应不同的环境。滑动窗口法的基本思想是在机器人当前位置和目标位置之间建立一系列重叠的滑动窗口，通过窗口内的路径规划得到机器人的局部路径，然后将不同窗口内的路径拼接起来得到机器人的全局路径。

2.5.2 全局规划

全局规划算法分为启发式算法和人工智能算法。

1. 启发式算法

Dijkstra 算法：Dijkstra 算法由 E.W. Dijkstra 在 1959 年提出。它是解决有向图中最短路径问题的典型最短路径算法。它的主要特点是以起点为中心延伸到终点。图的每条边由两个顶点组成一个有序元素对。边的值由权重函数描述。该算法维护两个顶点集 A 和 B。初始集 A 为空。每次将 B 中的一个顶点移动到 A，并且选择的顶点确保从起点到该点的所有边权重之和最小化。因为算法需要遍历更多的节点，所以效率不高。

A* 算法：哈特等人 [19] 在 1968 年提出了 A* 算法。A* 算法是在 Dijkstra 算法的基础上发展起来的。从特定节点开始，更新当前子节点的加权值，以加权值最小的子节点更新当前节点，直到遍历完所有节点。A* 算法的关键是建立评估函数 $f(n)$, $f(n) = g(n) + h(n)$, 其中 $g(n)$ 表示从初始节点到节点 n 的实际成本, $h(n)$ 表示状态空间中从节点 n 到目标节点的最优路径的估计成本。两个节点之间的欧几里得距离通常取为 $h(n)$ 的值。当 $g(n)$ 的值不变时, $f(n)$ 的值主要受 $h(n)$ 的值的影响。当节点靠近目标节点时, $h(n)$ 的值较小, $f(n)$ 的值相对较小。结果, 它保证了对最短路径的搜索总是在目标点的方向上进行。A* 算法考虑移动机器人目标点的位置信息, 沿着目标点进行搜索。与 Dijkstra 算法相比, A* 算法的路径搜索效率更高。

D* 算法：A* 算法主要用于静态环境的全局搜索。然而, 实际应用中移动机器人的路径规划是逐渐感知环境信息的, 并且是动态的。Stentz 在 1994 年提出了 D* 算法, 主要用于机器人探索路径。D* 算法的问题空间表示为一系列状态, 状态代表机器人位置的方向。D* 算法的原理与 D* 算法基本相同, 都是用 arc 的代价来保证搜索的方向。此外, 一些学者研究了 D* 算法, 如场 D* 算法和 Theta* 算法。

2. 人工智能算法

人工智能算法目前有人工神经网络法（ANN）、遗传算法（GA）、蚁群优化算法（ACO）、粒子群优化算法（PSO）、模拟退火算法（SA）等

2.6 定位模块

移动机器人导航系统需要实时对自身处于环境中的位置和姿态进行更新，无论是规划还是避障都要求移动机器人对自身位姿的精确掌握。定位信息不仅仅是需要知道机器人在当前环境中的所处位置，更重要的是移动机器人导航系统需要根据当前朝向等信息，及时调整位姿，以应对可能到来的碰撞和运动执行指令。目前广泛应用于移动机器人导航系统的定位方式有：1. 基于 GPS 的卫星导航定位法，定位精度高，应用广泛，缺点是成本高、卫星定位无法用于室内和遮蔽严重场景；2. 利用加速度计和陀螺仪等传感器测量机器人运动状态的惯性导航定位，成本低廉但是会存在漂移和累计误差；3. 激光里程计和视觉里程计，优点是成本可控，只需要一个激光雷达或者摄像头，缺点是对算法的要求较高，后端优化计算资源消耗大；4. 车轮硬件里程计定位，利用轮子之间安装的光电里程计定位，成本低廉、效果明显，但是累计误差较大，一般只用来做辅助定位。以上定位方法中，方法 1 和方法 3 是实际系统中使用最多的主定位方法，卫星定位方法主流应用于园区移动机器人定位、无人驾驶车辆定位，并且随着技术手段丰富，目前的高精度车载组合导航定位模块定位精度高、成本低、且数据丰富多元，满足各种需要；而激光雷达里程计和视觉里程计的方法随着算法的成熟与大算力芯片的普遍应用，商业化应用也日趋成熟与稳定。

2.6.1 高精度车载组合导航定位模块

高精度车载组合导航定位模块又称为 RTK(Real-Time Kinematic)，即以实时动态载波相位差分技术为基础的高精度 GPS 定位技术，因多在内部集成 IMU，又可以测量相应的加速度等信息，通常定位精度可达亚厘米级，故称为高精度车载组合定位模块。

1. RTK 定位流程

RTK 定位流程主要如下：1. 建立基准站在开始 RTK 定位之前，需要先建立一个基准站。基准站需要安装在已知位置，并与 GPS 卫星接收机连接。基准站接收卫星信号后，可以通过精确的测量手段，如全站仪、激光测距仪等，确定其准确的位置坐标，并记录下来。

2. 信号接收移动站接收卫星信号后，将接收到的信号传递给 GPS 接收机。接收机通过测量接收到卫星信号的时间差，可以计算出卫星信号传播的距离。

3. 差分处理基准站和移动站之间的距离差异会导致定位误差。差分处理的目的就是消除这些误差，从而提高 RTK 定位的精度。差分处理的具体步骤如下：

3.1 基准站记录卫星信号距离；基准站接收卫星信号后，记录下每颗卫星信号到基准站的距离。

3.2 基准站计算修正值；基准站通过比较接收到的卫星信号距离和实际距离，计算出每颗卫星信号的误差修正值。这些修正值包括大气延迟、多径效应、钟差等误差。

3.3 移动站接收修正值；基准站将误差修正值通过无线电信号发送给移动站。移动站接收到这些修正值后，将其应用到自身测量的卫星信号距离值上进行修正，从而消除定位误差。

4. 计算位置；移动站接收到修正值后，通过计算移动站接收到的卫星信号距离和修正值，计算出移动站的位置坐标。移动站的位置坐标与基准站的位置坐标之间的距离差异可以用来计算出其精确的位置坐标。

2. 坐标系转换

上述流程拿到了 RTK 的定位信息是经纬度和高程信息，并不能直接使用于移动机器人定位。需要将世界大地坐标系（WGS 84）坐标系信息转换成为以地球质心为坐标系原点的球心坐标系，再由球心坐标系转成以 RTK 所在系统为站心的站心坐标系（东北天 ENU 或北东地 NED）。

如图所示，其中椭球体表示地球，红色坐标系表示当前 RTK 所处的站心位置 P，原点到 P 点的距离为 r ，原点到点 P 的连线与正 z-轴之间的天顶角为 θ ，以及原点到点 P 的连线，在 xy-平面的投影线，与正 x-轴之间的方位角为 φ

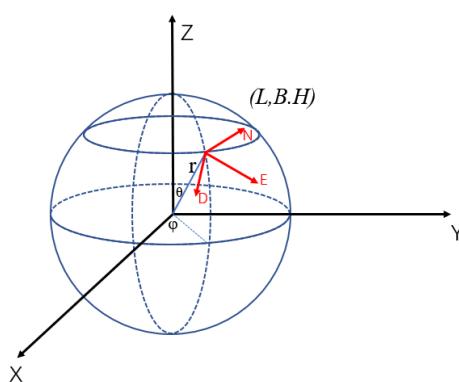


图 2.4 各种坐标系示意图

WGS 84 大地坐标系 → 球心坐标系：WGS84 是一种大地坐标系，使用经度、纬度和海拔高度来表示地球上的位置。WGS84 的坐标系原点位于地球质心处，因此需要将大地坐标系转换为球心坐标系来进行地球上点的计算。

球心坐标系是一种三维笛卡尔坐标系，它使用 X、Y、Z 坐标轴表示点在三个方向上的位置。球心坐标系的原点位于地球质心，因此所有地球上的点都可以用 X、Y、Z 坐标来表示。

转换大地坐标系到球心坐标系的公式如下：

$$\begin{aligned} X &= (N + H) * \cos B * \cos L \\ Y &= (N + H) * \cos B * \sin L \\ Z &= [N * (1 - e^2) + H] * \sin B \end{aligned} \quad (2.4)$$

其中，B 是纬度，L 是经度，H 是海拔高度，N 是地球半径在该点的半径， e^2 是椭球的离心率的平方，其值为 0.00669437999014。X、Y、Z 分别表示球心坐标系中点的三个方向上的坐标。H 是点的海拔高度，表示点相对于地球表面的高度。N 是地球半径在该点的半径，表示从地球中心到该点的距离。 e^2 是椭球的离心率的平方，用于计算 N。

球心坐标系 → 站心坐标系 NED：NED 坐标系：North-East-Down，简写为 NED。

$$\begin{aligned} N &= -\sin(B) * \cos(L) * X_0 - \sin(B) * \sin(L) * Y_0 + \cos(B) * Z_0 \\ E &= -\sin(L) * X_0 + \cos(L) * Y_0 \\ D &= -\cos(B) * \cos(L) * X_0 - \cos(B) * \sin(L) * Y_0 - \sin(B) * Z_0 \end{aligned} \quad (2.5)$$

其中，B 是纬度，L 是经度， X_0, Y_0, Z_0 分别表示球心坐标系原点相对于参考点的位置坐标。

经过坐标系转换的点已经是理想条件下 RTK 系统所处的站心坐标系下的位置信息，再与 RTK 中得到的点的 roll、yaw、pitch 三项结合，就可以作为机器人系统的位姿信息使用，参与后续计算中的位姿变换等。

2.6.2 激光里程计和视觉里程计

激光里程计定位（Laser Odometry and Mapping，LOAM）和视觉里程计定位（Visual Odometry，VO）是两种常用的基于传感器数据的定位方法，它们可以用于机器人、自动驾驶等应用中。激光里程计定位方法使用激光雷达作为传感器，通过激光点云数据来计算机器人的运动和位置。在 LOAM 中，激光雷达发射激光束并接收反射回来的光，生成点云数据。然后，通过分析这些点云数据，可以计算出机器人的位置和姿态。视觉里程计定位方法使用相机作为传感器，通过相机图像来计算机器人的运动和位置。在 VO 中，通过相邻帧之间的图像差异，来计算机器人的相对运动。然后，通过相机内参和外参等参数，将相对运动转化为机器人在三维空间中的运动。相较而言，激光里程计定位方法的优点是精度高、

鲁棒性强、适用于复杂环境。视觉里程计定位方法的优点是设备便携、成本低、适用于室内场景。然而，视觉里程计定位对光线、噪声等环境条件要求更高，激光里程计定位则对计算资源要求更高。通常对这两者加以结合，扬长避短，获得不错的定位效果。通常，激光里程计和视觉里程计会应用于 SLAM 系统中，近些年来，出现了相当一批具有代表性的工作，视觉领域中，Mur-Artal et al.^[31]提出了一种基于特征点的稠密 SLAM 系统，通过在图像中提取 ORB 特征点，并使用 RANSAC 等算法来计算相机运动和场景深度信息。该系统具有实时性、准确性和鲁棒性等优点，在室内和室外场景中都能够进行定位和地图构建。并且于 2020 年提出了 ORB-SLAM2 的改进版本，增加了对相机、地图再定位的支持。Qin et al.^[32]提出了一种基于单目相机和 IMU 的 VIO 系统，称为 VINS-Mono。该系统采用了一个可扩展的状态估计器，可以实现高精度、低延迟的定位。而在激光领域中，前面提到的^[24]是一种基于激光雷达的稠密 SLAM 系统，通过使用分段扫描匹配和点云配准等算法，来计算机器人的运动和位置。该系统具有高精度、实时性和鲁棒性等优点，在大型、复杂环境中进行定位和地图构建。

视觉和激光的融合领域亦有突出的工作，Wisth et al.^[33]提出了一种基于融合激光雷达、视觉和惯性传感器的紧耦合定位方法，称为 LVI-Odometry。在该方法中，激光雷达、视觉和惯性传感器互相协同，实现了高精度的实时定位。LVI-Odometry 将 SLAM 问题转化为多模态地标跟踪问题，通过优化地标的状态估计，从而实现机器人的位姿估计。

2.7 运动控制

移动机器人领域最常用的运动控制方法主要有 6 种，每一种有具有其独特性质。

PID 控制器：PID 控制器是一种基于反馈的控制器，可以对机器人的位置、速度、加速度等运动状态进行控制。PID 控制器的优点是简单易实现，可以通过调整参数来满足不同的运动要求，但在一些复杂的控制任务中，需要较为复杂的控制算法。

路径追踪控制：路径追踪控制是一种基于轨迹的控制方法，它可以根据预设的轨迹来控制机器人的运动，以达到预期的运动效果。路径追踪控制的优点是可以精确控制机器人的运动轨迹，但需要对机器人和环境进行精确建模。

动力学控制：动力学控制是一种基于物理模型的控制方法，它通过建立机器人的动力学模型，对机器人进行运动控制。动力学控制的优点是可以考虑机器人的动态特性，能够处理高速运动和非线性系统的控制问题，但需要对机器人的动力学进行精确建模。

人工势场控制：人工势场控制是一种基于势场的控制方法，它通过建立机器人和环境的势场，控制机器人沿着势场梯度方向运动。人工势场控制的优点是简单易实现，能够实现障碍物的避障和动态避障，但容易出现局部最小值陷阱问题。

模糊控制：模糊控制是一种基于模糊逻辑的控制方法，它通过将模糊逻辑应用于机器人的控制系统中，对机器人进行运动控制。模糊控制的优点是可以处理模糊和不确定性的问题，但需要进行大量的经验调试。

强化学习：强化学习是一种基于试错学习的控制方法，它通过与环境交互来学习最优的行动策略。强化学习的优点是能够处理复杂的非线性系统和未知环境下的控制问题，但需要大量的训练时间和计算资源。

2.8 局部避障

移动机器人的安全稳定性很大程度上需要靠导航系统的避障功能来保障。通过传感器对周边环境中障碍物的感知，确定自身与障碍物距离，并且通过算法避开障碍物的区域。这种避障是可以轻易做到的，但是移动机器人不仅仅是面临当前避障的考验，还要面临相应的避障之后的移动规划问题。故相应的避障策略必须有所调整。

ORCA (Optimal Reciprocal Collision Avoidance)：是一种局部避障方法，它是一种基于动态几何的方法，通过计算机模拟人类在复杂环境中避免碰撞的行为来实现避障。它主要用于多智能体系统中的避障问题，如机器人团队协作、无人机编队等。

ORCA 算法的基本思路是将每个机器人视为圆形，然后利用圆形的运动学约束计算机器人的速度范围，再根据其周围的障碍物信息计算机器人的速度矢量，以避免与周围机器人发生碰撞。ORCA 算法的优点是计算简单、易于实现，适用于高密度机器人交通场景。

ORCA 算法的缺点是不能处理动态障碍物，即不适用于环境中出现突然障碍物的情况。此外，ORCA 算法存在局限性，可能会导致机器人进入局部最小化陷阱（Local Minima Trap）或者被卡在障碍物周围无法移动。因此，在实际应用中，需要根据具体的场景进行优化和改进。

MPC (Model Predictive Control)：也常常运用于移动机器人导航系统的避障功能中，在 MPC 中，避障问题通常被看作是一种约束条件，通过将避障约束加入到控制器中，来实现机器人的安全移动。在实现中，可以通过在 MPC 控制器中引入障碍物的状态信息和避障目标函数，以指导机器人的运动轨迹规划。具体来说，MPC 避障方法的基本步骤包括：

建立机器人的动力学模型和环境模型，包括机器人的位置、速度、加速度等状态信息，以及环境中的障碍物信息。

利用动力学模型和环境模型，在预测时间窗口内计算机器人的轨迹，以及与障碍物的相对位置。

将避障目标函数引入到 MPC 控制器中，例如最小化机器人与障碍物的距离，或者最大化机器人可行的运动范围。

在实时控制中，通过调节控制器的参数，来实现机器人的避障和安全移动。

MPC 避障方法具有灵活性和可扩展性，适用于不同类型的移动机器人，如无人车、无人机等。同时，MPC 避障方法可以考虑到机器人的动态特性和环境的不确定性，具有更好的鲁棒性和性能。

第3章 多层级导航系统方法研究与算法设计方案

3.1 引言

底层导航原引言：

本实验室的智能移动机器人的导航系统主要分为两个层级，根据系统层、信息决策层与硬件系统的信息交互，人为划分成为两个层级，本文把与硬件系统，例如阿克曼底盘以及、激光雷达、车载组合定位模块有信息交互的导航层划分为底层导航系统。底层导航系统在系统中扮演的角色主要是根据高层导航系统的规划，将高层导航系统规划得到的一系列目标点，拆分为移动过程中的一一个个目标，在局部规划的过程中，实时地避障。底层导航地创新之处在于解决了底层导航中常出现的车辆整体陷入死胡同的尴尬境地，避免了车辆在避障过程中由于算法不成熟导致的卡死问题，在一定程度上支持了移动智能机器人在园区场景下的长期生存难题。

底层导航系统内部集成有多个子模块，它作为多层次导航系统的子系统，同时也可以自成体系。底层导航的运行，需要对周围环境的感知，也就是根据传感器的采集数据，经过一定算法处理，生成特定的环境信息，也就是得到的局部点云信息；得到了周围环境信息之后，对信息中的数据进行特定化的处理，要让移动智能机器人的计算核心理解各种环境信息，比如可通行的区域与环境中存在的障碍物；其次，需要设计一个局部规划器，规划器要充分考虑移动机器人的底盘模型，并结合避障系统，对局部的路径进行规划。最后，规划出的路径以控制信号形式发送给执行部件——也就是底盘，去做实际的动作执行。

高层导航原引言：

上述章节是在没有全局先验地图情况下对目标点进行追踪，并且在追踪过程中进行避障等操作。当移动机器人具备了局部情况下的追踪与避障功能之后，这种基于局部点云的算法只适合在小范围内进行导航，例如在学校的宿舍区或者食堂区域内局部活动。当给定的目标点距离过远，超过局部点云的承载极限时（规划的目标点在点云之外），又或者给定的目标点与移动智能机器人之间存在不可越过障碍物时，例如园区内河、湖泊和小山等，移动机器人无法得到障碍物后面的情况，只依赖当前局部规划算法的逻辑，根据给定目标点的朝向和当前移动机器人的角度来单纯地局部规划，那么移动机器人便会陷入局部最优之后——以局部算法的最优解去不停地尝试解决问题。这边会导致移动机器人始终无法到达目标点。导航任务因此搁浅。

所以一旦涉及到园区的全局场景，上述底层导航功能便不再适用，当然一个可行的方法是在人为事先采集好路点的情况下进行发布，并且由底层导航接收

而后完成任务。那么为了弥补这一导航系统的缺陷，势必需要对全局园区环境进行了解，并且在具有全局先验信息的情况下，直接进行地图层级的规划，并且将规划得到的路径点逐一发布给底层导航，由此完成整个导航任务。完成地图层级地规划需要几个先决条件，一是构建的全局地图，用来存储全局的路径信息以及障地貌信息等；二是基于地图的路径规划算法，在地图上规划处位于两个点之间的可行驶路径，而后将规划得到的路径点发布出去。

3.2 基于全局地图的高层导航方法

3.2.1 基于 RTK 的多层地图构建

目前的建图算法十分多见，有基于 SLAM 系统的实时定位建图算法，可以在算法运行的同时，实时地将地图建立起来，效率相比其他算法较高；也有基于全球定位系统、卫星定位的离线建图方法。两种方法是目前比较主流的建图算法，且两者相比具有不同的优缺点。前者效率较高，无需额外的定位设备，只需要激光雷达或者摄像头，比较典型的是 LeGO_LOAM 和 ORB-SLAM 为代表的 SLAM 算法。同时 SLAM 方法无需后期的人工拼接，但是在建图的精度上不如后者；而基于卫星定位建图算法主要依赖是外部的定位设备，需要额外的成本，同时必须进行人工后期拼接才能使用；但是它的优点也是极其明显的：可以人工拼接并做后期的地图处理，同时具有较高的定位精度，也允许在一定程度上对地图进行编辑并在特定的算法需求上提供不同的处理方案。在本实验是移动智能机器人的构建过程中，采购了中海达的高精度车载组合定位模块，并且由于园区场景下，本着一次建图、永久使用的原则，无需重复多次建图，因此综合建图精度、成本控制以及使用需求等多方面因素，最终决定使用基于高精度车载组合定位系统（以下简称 RTK）的方案进行园区场景的地图构建。

1. 点云地图

为了建立多层次的全局地图，首先需要建立的是园区下的全局点云地图。点云的构建过程是对采集到的点云按照位姿进行拼接以及对拼接后的地图进行后期处理的过程。为了在校园场景下构建一幅适合移动智能机器人的导航系统，针对各个可供移动智能机器人通过的路径以及走廊等位置进行数据采集，主要的采集平台是煜禾森公司的 FR-07 智能机器人底盘与装配的 Velodyne 公司的 VLP-16 型激光雷达。关于此平台的参数将会在具体的硬件章节做更加深入的介绍。

在使用设备采集相应的点云信息之前，需要对车身以及设备之间的坐标系进行标定。标定后的设备共有三大主要坐标系，分别是移动智能机器人底盘后轮中心的 base_link 坐标系、VLP-16 激光雷达坐标系以及 RTK 坐标系。各个坐标系之间的相对位置关系如下图所示：



图 3.1 模板图片

首先在园区全局内选定一处作为园区地图内的世界坐标系原点，考虑到整个地图的合理性，选定的坐标系原点为校园中心位置。此处参考的地心坐标系统是 WGS 84 世界大地坐标系。后续采集的点云数据均据此作为坐标系原点与位姿基准展开数据处理。

整个构建点云地图的流程框图如下所示：



图 3.2 模板图片

采集的软件系统为 Robot Operating System (简称 ROS 系统)。采集地图的命令为： sudo record -a；

(1) 地面提取

地面提取的方式有多种，目前主流应用的是直接计算法、平面拟合法以及滤除法。其中直接计算法是直接计算相邻两条激光线的俯仰角，俯仰角变化在一定范围内的可以被认为是地面点；平面拟合法在底层导航一章中已经有叙述，主要

是通过选取种子点拟合出相应平面，并且通过迭代方法使得地面选取更加精准。再全局地图的构建过程中，由于不再是之前简单将地面点去除，只需要迭代出地面后去除即可，关注点在于非地面点。此时我们对于点云中地面点的质量有了更高的要求，是“宁缺毋滥”地筛选符合要求地地面点，人为增加了点的筛选门槛。下面给出具体方法。对采集得到的点云包进行分析，找到其中的 velodyne_points 话题，对于包中所含有的任意一帧点云进行分析，提取其中的非地面点部分与地面部分。提取地面部分点云的方法如下所示：对于激光雷达采到的任意一点，此点与激光雷达的位置关如下图所示：



图 3.3 模板图片

在激光雷达坐标系中， θ 表示点与水平 xoy 平面的夹角， x 、 y 、 z 分别表示到坐标轴的距离首先根据公式

$$radius = \sqrt{x^2 + y^2} \quad (3.1)$$

计算点的采集半径 radius，再根据

$$\theta = atan2(z, radius) * 180/\pi \quad (3.2)$$

计算得到点与水平面的夹角，atan2 函数的象限划分与对应角度不同于 atan 函数，其原理为

$$atan2(y, x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & x > 0 \\ \arctan\left(\frac{y}{x}\right) + \pi & y \geq 0, x < 0 \\ \arctan\left(\frac{y}{x}\right) - \pi & y < 0, x < 0 \\ +\frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ \text{undefined} & y = 0, x = 0 \end{cases} \quad (3.3)$$

低于激光雷达坐标系 xoy 平面的点角度判定为 $-$ ，而高于此平面的点角度判定则为 $+$ 。得到了某点的角度 θ ，根据 VLP-16 激光雷达的角度分布，便可以得到某点所属于的扫描线序号 scanID，根据地面扫描点主要集中于最下面的三条扫描线的特点，只取角度为 -15° 、 -13° 、 -11° 的三条扫描线上的点进行分析。根据 scanID 将属于某条扫描线的点集合起来，检测点的粗糙值，粗糙值的计算公式为

$$c_i = \frac{1}{|P|} \sum_{j \in P, j \neq i} \|p_i - p_j\| \quad (3.4)$$

表达式的意思是首先取当前点同 scanID 的前后各十个点， c_i 表示当前点的平滑度， P 表示当前点与其前后十点的集合， $|P|$ 值为邻点的数量， p_i 、 p_j 分别表示当前点的向量与其邻点的向量，对向量取 2-范数。最终的平滑度表示为当前点与其周围——同一扫描线前后各 10 点之间的距离均值。并且在最后对下三线上的点完成分析之后，设置相应阈值，使得地面点合格率约为 $1/3$ 左右。最终，通过此方法取得的地面点在保证了点密度的情况下最大限度保证了地面点的质量。

(2) 基于 RTK 的地图合成

每一帧地面点云地数据拿到之后，需要对地面点云根据位姿进行拼接，拼接点云位姿来源于数据采集过程中同时间戳录制的 RTK 位姿数据。在前面 fast_gicp 的叙述中也有提到两帧不同位姿的数据如何进行拼接，其实就是统一坐标系的过程。与之前不同的是，之前的拼接没有一个全局的参考坐标系，是对车身位置实时位姿变换。当在园区场景内选定了全局坐标系之后，若某一帧点云的位姿相对于全局位姿的变换为 R 、 t ，分别表示点云帧相对于全局位姿所作的旋转与平移。任意时刻，某点与当前坐标系的位置示意图如下所示。将当前点云中某



图 3.4 模板图片

点变换到全局坐标系下的变换为

$$P'_1 = \mathbf{R} * P_1 + t \quad (3.5)$$

P_1 表示点云的中某点的向量 $[x, y, z]^T$, 当前位姿, P'_1 表示变换到全局坐标系下的位姿 $[x', y', z']^T$ 。该式还可以通过齐次坐标和变换矩阵的形式表现

$$\begin{aligned} P'_2 &= \mathbf{T} * P_2 \\ \mathbf{T} &= \begin{bmatrix} \mathbf{R} & t \\ 0^T & 1 \end{bmatrix} \end{aligned} \quad (3.6)$$

P_2 表示点云的中某点的向量 $[x, y, z, 1]^T$, 当前位姿, P'_2 表示变换到全局坐标系下的位姿 $[x', y', z', 1]^T$ 。

此处可以直接调用 `pcl::transformPointCloud` 将一帧地面点云位姿变换到全局坐标系下, 此过程仍然需要在一定帧数拼接完成后进行降采样操作。操作原理与上一章类似。至此, 地面点云的点云地图拼接完成, 效果如下所示。

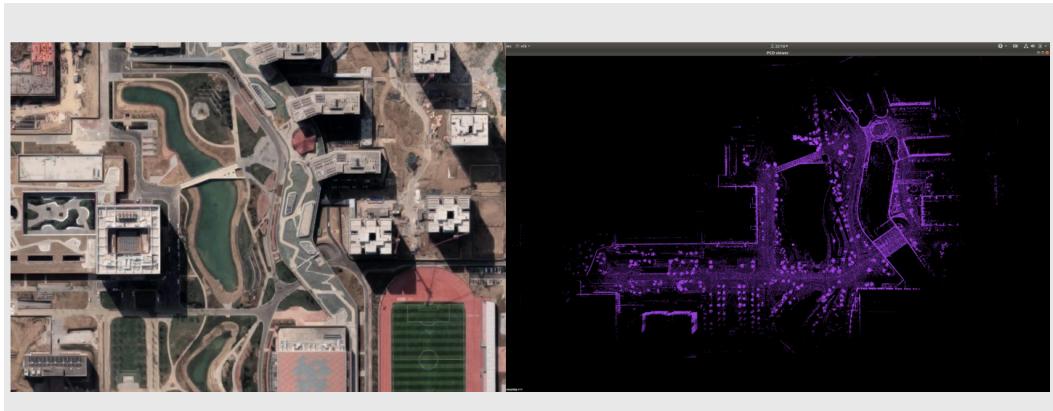


图 3.5 全局点云地图与原图片对比

可以添加一部分用来讨论 WSG84 到 RTK 站心坐标系的转换

2. 栅格地图

在得到了上述的地面点云地图 `roadMapAll` 之后, 开始需要对这些可行驶区域进行栅格化。在实际使用过程中, 栅格化不必要对整个点云地图进行, 只需要对需要的可行驶区域部分, 也就是上述得到的地面点云地图进行栅格化操作。第一步获得此 `roadMapAll` 中点的极限值——所有点中最小的 x 值, y 值, z 值以及最大的 x 值, y 值, z 值。依据此值对整个 `roadMapAll` 区域进行划分, 按照 $0.5*0.5m$ 的分辨率得到相应的栅格地图, 每一个方格内所有点的平均高度为 `gridAverageHeight`;

(1) 泛洪算法

接下来使用著名的泛洪算法中的四邻域泛洪法,

(2) 腐蚀算法

腐蚀算法主要目的是刻蚀掉孤悬于主区域之外的零星可行驶区域，此区域与其他可行驶区域的连接性不强，我们是不希望它作为后续路径规划的节点网格的。

3.2.2 基于距离地图的全局路径规划算法

在选定的栅格地图基础上，进行全局的路径规划，规划时不考虑全局地图上的动态性，默认全局地图上的可行驶区域在规划时刻均为静态可通行。一般来说，全局规划的算法有很多种，例如 RRT、RRT*、A*、Hybird A*、D*、Dijkstra 等等。其中 RRT 簇适用于随机规划路径，但规划出的路径不是最优的；A* 簇算法适用于具备全局地图情况下的规划，并且规划出的路径是最优路径，其衍生算法 D* 可以在地图有改变的情况下进行局部重规划；Dijkstra 算法是一种 BFS 式的搜索算法，与 A* 类算法相比，具有较高的搜索代价。由于我们提前说明不考虑全局地图上的动态性，故在全局搜索中使用的是 A* 规划算法。

1. A* 算法

原理简述：A 星算法是一种常用于寻路问题的算法，可以在图形图像中找到从一个起点到一个目标点的最短路径。一共如下 3 大步骤：初始化起点和终点：首先将起点加入 open 列表，open 列表存放所有待考虑的节点。同时初始化 g 值和 h 值，其中 g 值是起点到当前节点的距离，h 值是当前节点到终点的估计距离（如欧几里得距离）。

循环直到找到终点：在每次循环中，从 open 列表中选取 f 值最小的节点作为当前节点，并将其移入 closed 列表中。如果当前节点为终点，则搜索完成。否则，将当前节点的邻居节点（即可以直接到达的节点）加入 open 列表中，计算它们的 g 和 h 值。如果邻居节点已经在 closed 列表中，或者 g 值更大，则不做处理。否则，更新 g 值，重新计算 f 值。

最终路径的生成：当终点被找到时，从终点开始按照每个节点的父节点反向寻找路径，直到回到起点为止。这就是从起点到终点的最短路径。A 星算法的优点在于可以通过启发式函数对估计的距离进行优化，使得算法能够快速找到最短路径。同时，算法能够在不考虑地形的情况下进行路径规划，这使得它在游戏设计、机器人路径规划等领域得到了广泛应用。

A* 算法的问题可以描述为

如何穿过避开障碍物并且在最小消耗的情况下到达终点位置。

在 A* 算法中最重要的是启发函数的设置，通常规律下，A* 算法的启发函数可以描述为

$$F = \alpha G + \beta H \quad (3.7)$$



图 3.6 模板图片

其中， F 表示总代价

G = 从起点 A 移动到指定方格的移动代价，沿着到达该方格而生成的路径。
 H = 从指定的方格移动到终点 B 的估算成本。这个通常被称为试探法，因为这是个猜测。直到我们找到了路径我们才会知道真正的距离，因为途中有障碍物阻挡于当前网格节点与终点之间。通过调节 G 与 H 的系数 α 、 β 可以对搜索的方式进行调整。例如当在距离地图中希望规划出的路径远离地图场景中本就存在的障碍物时，便可以将 H 调整为到终点的估算成本与周围障碍物距离的加权和。

3.3 基于局部点云的底层导航方法研究

3.3.1 基于 Fast-GICP 的局部点云

导航系统的主要感知设备为 Velodyne 的 VLP-16 型号的激光雷达，雷达的主要参数为：测量距离 100m， 360° 水平视场角， $\pm 15^\circ$ 的垂直视场、竖直分辨率 2° 、水平分辨率在实验室移动机器人上 10Hz 下为 0.2° 。其中由于激光线束为 16 条，所以 10Hz 下扫描一圈的单线采集点数为 1800 个激光点。这在园区场景下来说是十分稀疏的，导致在后面的避障过程中无法有效对相应地点云进行信息处理，于是需要使用点云的技术拼接手段，将某个范围内的点云进行融合，形成以车身为圆心的一定范围内的稠密局部点云。此部分使用 fast-gicp 为算法主体，在其基础上，加入对点云帧的质心位置评估，使得算法整体的实时性和可用性在实际场景中获得大幅度提升，免去了对很多边缘离群点的计算量。

不同于全局地图中使用 RTK 获取移动机器人位姿，在底层导航的设计中，兼容了室内场景的应用模式，使用 Fast-GICP 可以在室内进行应用。

1. 车身当前位置范围内点云帧选取

为了得到当前时刻车身周围的稠密局部点云，需要保存车身圆周范围一定数量的点云帧，评价某个时刻采集到的点云数据 P_t 与当前车身位置的距离，需要获取 P_t 的质心，点云质心指的是点云的质量中心，通常被认为是点云的质量集中于此点的假想点。质心的计算公式为：

$$\begin{cases} \bar{x} = \frac{\sum_{i=1}^N x_i}{N} \\ \bar{y} = \frac{\sum_{i=1}^N y_i}{N} \\ \bar{z} = \frac{\sum_{i=1}^N z_i}{N} \end{cases} \quad (3.8)$$

其中 \bar{x} 、 \bar{y} 、 \bar{z} 分别表示此帧点云质心的坐标数值。同时为了避免边缘噪点对于质心的计算影响，同时我们关注的点云也只限于车身范围内的 5m 半径内，所以通常会将点云基于其采样中心的半径 10m 计算其质心，这样既省去了计算量，由去除了采样帧远方边缘噪点的影响。事实上，在 PCL 库的内部已经集成了的相应的求取点云质心的函数 `pcl::compute3DCentroid`，只需要将相应的库加入文件中便可以调用。选取车身当前范围的一定数量的关键帧方式如下，首先初始化，根据时间顺序，选取一定数量的点云帧，完成初始化之后，基于位置信息的二维距离选取下一帧，并在移动的过程中，逐步替换掉最早的点云帧，保证点云池内存在的点云帧数量保持为定值，点云帧的选取示意图如下：图中红色点表示当前

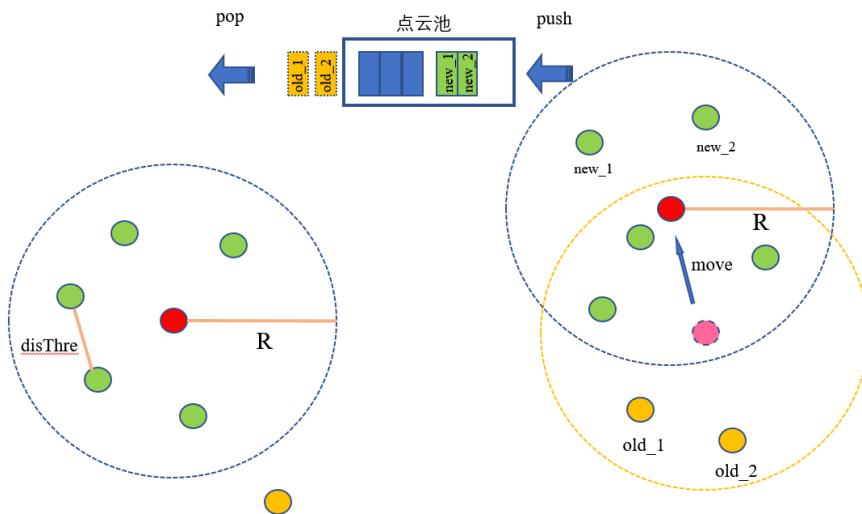


图 3.7 临近点云帧的选取

车身所处位置，绿色、橙色表示车身范围内与外的邻近点云质心所处位置。点云池内部的点的数量除了初始化时，始终保持数量不变，并且绿色点之间的距离有

最小阈值 `disThre` 限制。

2. Fast-GICP 点云拼接

点云池内具有一定数量的点云帧之后，对点云池内的点云进行拼接操作。点云池内的点云与原采集到的点云保持对应关系，为保证 ICP 算法的精准度，将原始帧输入 `fast_gicp` 处理模块，由此得到相应帧之间的位姿关系，并且位姿关系对上述帧进行拼接。

(1) Fast-GICP 配准算法

Fast Global Iterative Closest Point (Fast-GICP) 是一种用于点云配准的算法，是 Global Iterative Closest Point (GICP) 的一种改进。它通过使用局部高斯表达来计算点云间的相似性，并通过加速矩阵求逆和构建雅可比矩阵的方式提高了配准的速度和精度。Fast-GICP 的原理基本上是在 GICP 的基础上进行的改进。在 GICP 中，点云的对应关系是通过点之间的欧氏距离计算的，而 Fast-GICP 则采用了点云之间的局部高斯表达，这有助于解决欧氏距离在存在噪声或离群点时产生的误差问题。Fast-GICP 还使用了一些技巧来加速矩阵求逆和构建雅可比矩阵，例如采用 SVD 分解来避免矩阵求逆的计算和采用块状矩阵来优化雅可比矩阵的构建。这些技巧有助于提高配准的速度和精度，并使 Fast-GICP 成为一个非常有效的点云配准算法。其算法步骤为：

1. 定义目标点云和源点云 $Target$ 和 $Source$ ，其中 $Target$ 是固定点云， $Source$ 是需要变换以匹配 P 的点云。
2. 初始化初始变换矩阵 T_0 ，通常情况下可以将变换矩阵初始化为单位矩阵，以便于进行迭代，即 $T_0 = I$ 。
3. 对于每次迭代：
 - a. 通过应用当前变换矩阵 T ，将 $Source$ 变换到当前位置。
 - b. 计算 $Target$ 和 $Source$ 之间的最近点对，并计算最近点对之间的距离，距离计算公式为

$$d(p_i, q_j) = ||p_i - Rq_j - t|| \quad (3.9)$$

其中 p_i 表示点云 $Target$ 中的第 i 个点， q_j 表示点云 $Source$ 中的与 q_i 最近的点， R, t 分别表示当前变换矩阵 T 的分解处的旋转向量与平移向量。

c. 根据最近点对计算点的权重

$$\omega(p_i, q_j) = \exp\left(-\frac{d(p_i, q_j)^2}{2\sigma^2}\right) \quad (3.10)$$

其中 σ 是一个常数，用于控制高斯函数的宽度。

d. 计算均值向量

$$\mu_{Target} = \frac{1}{N} \sum_{i=1}^N p_i, \mu_{Source} = \frac{1}{N} \sum_{j=1}^N q_j \quad (3.11)$$

其中 N 表示点云中的点数。

e. 计算协方差矩阵

$$H = \sum_{i=1}^N w(p_i, q_j) \begin{bmatrix} q_j & 1 \end{bmatrix}^T \begin{bmatrix} p_i - \mu_P \\ 1 \end{bmatrix} \begin{bmatrix} p_i - \mu_P \\ 1 \end{bmatrix}^T \begin{bmatrix} q_j & 1 \end{bmatrix} \quad (3.12)$$

f. SVD 分解 H 矩阵

$$\begin{aligned} H &= U \sum V^T \\ R &= VU^T \end{aligned} \quad (3.13)$$

$$t = \mu_P - R\mu_Q$$

其中， U 和 V 是 H 的左奇异向量和右奇异向量， Σ 是 H 的奇异值矩阵。

d. 更新变换矩阵 $T = T_{corr}T$ ，并且更新点云 $Target$ 中的点。

$$p_i \leftarrow Rp_i + t \quad (3.14)$$

4. 如果达到停止条件，则返回最终变换矩阵 T ，否则返回步骤 3。

(2) 局部点云生成

Fast-GICP 作为算法主体生成局部点云，其流程图如下：初始化时，*Source* 点云是从点云池队列中进行读取，*Target* 点云则是采集车身当前时刻的点云，通过 Fast-GICP 算法配准后，合成 sub 局部点云地图。当参与构建局部点云的点云帧到达阈值 n (n 为点云池内点云帧总数)，初始化完成，第一个局部点云构建完成。当完成初始化之后，点云池队列中的点云数量更新达到阈值后，即当点云池内部点云帧数量更新第 $\frac{3n}{4}$ 帧时，说明此时小车距离上次生成局部地图的位置已经足够远，在运行则要离开当前局部点云的范围，此时更新局部点。

经过此步骤，拼接得到的局部点云效果图如下：

3. 稠密点云降采样

激光点云经过拼接后，同一物体处的扫描点迅速增加，影响后续算法的实时性。对得到的局部稠密点云进行适当的降采样，可以使得局部点云信息更加清晰，条理分明。得到的局部点云地图是经过点云配准算法的处理，其中多了冗余信息与噪点，直接处理计算量资源消耗巨大，因为降采样是必要的操作。

降采样的方法有多种，例如体素网格下采样、均匀下采样、几何曲率下采样、随机下采样等。这里选择的是体素网格下采样法。体素的概念就是将空间划分成一个个立体的方格，每一个方格就称为一个体素网格，降采样的思路是检查每个体素中是否有点存在，若哟，则用一个点代替体素内的点集，此处使用体素网格中心点坐标作为采样点。体素网格降采样的优点是可以通过控制体素网格的分辨率控制采样点之间的距离。

经过降采样后得到如下的降采样点云，对降采样算法的分辨率分别使用 resolution = 0.25、0.5、0.75，效果图分别如图 (a)、(b)、(c) 所示。

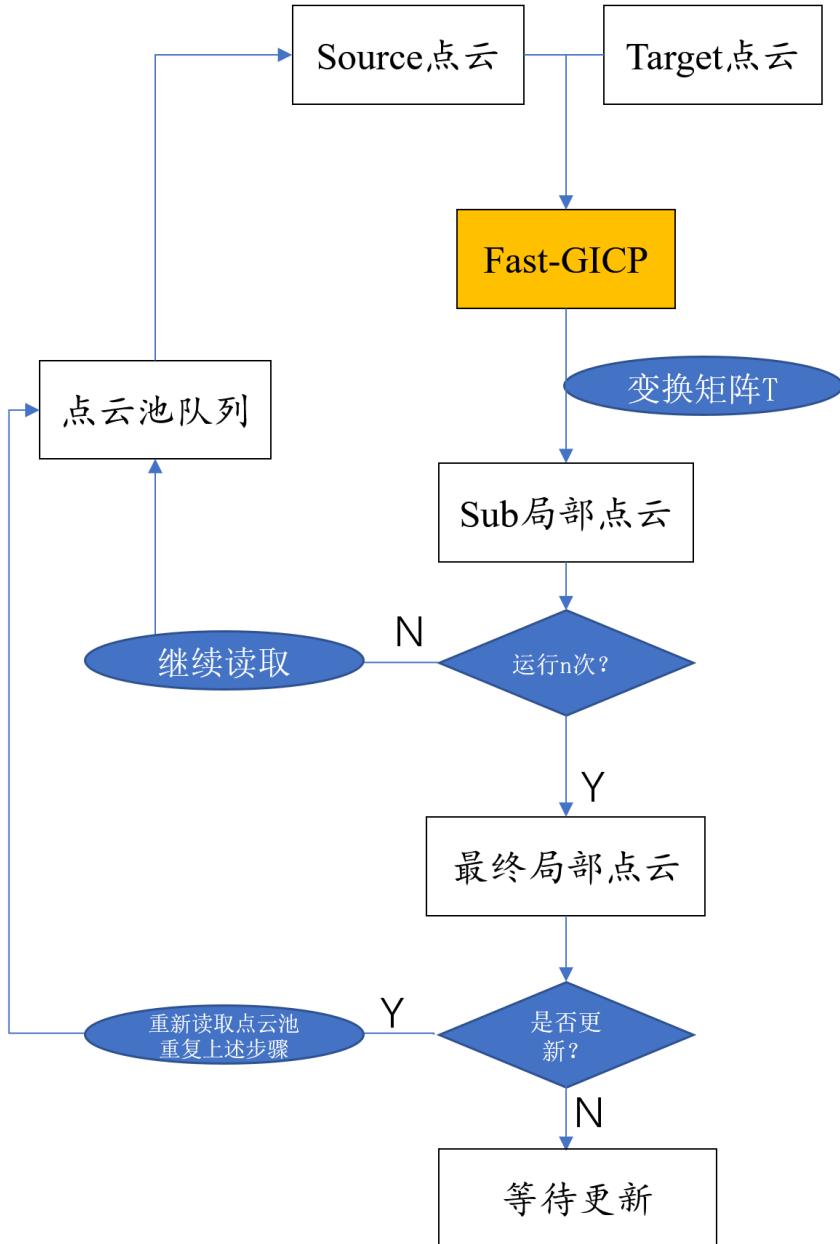


图 3.8 局部点云生成过程流程图

此步骤完成后，便可以得到以车身目前在环境中所处位置为基准的局部点云，根据此可以开展点云的后续处理工作。

3.3.2 局部点云的分割与聚类

对当前的局部点云进行有效的分割处理并聚类，是接下来进行避障以及路径规划的基础，点云的分割聚类主要是为了让底层导航系统在避障过程中识别障碍物，并且对动态障碍物信息做出实时的反馈与动作。具体的分割与聚类工作如下。

基于 Bogoslavskyi 在文章 Fast range image-based segmentation of sparse 3D

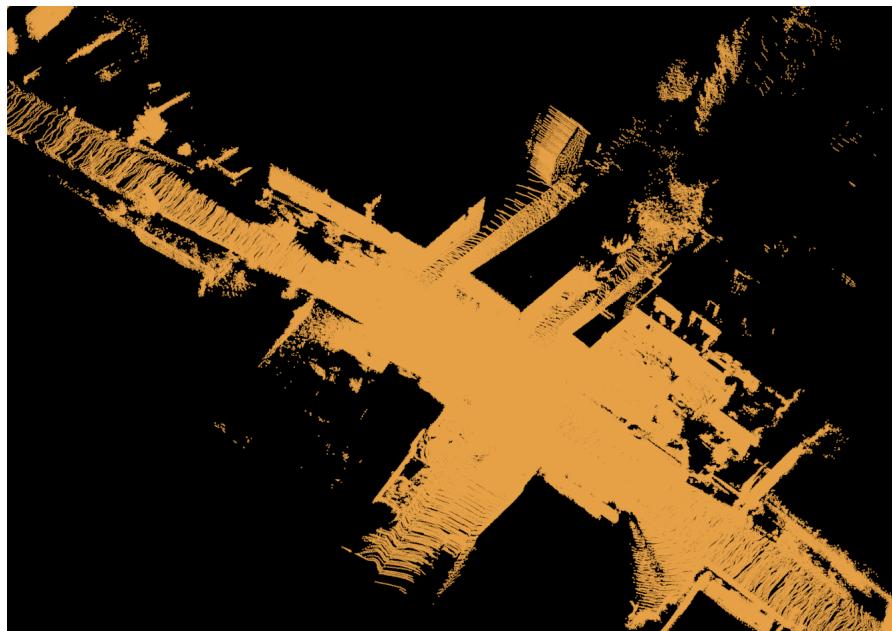


图 3.9 经过拼接的初始局部点云

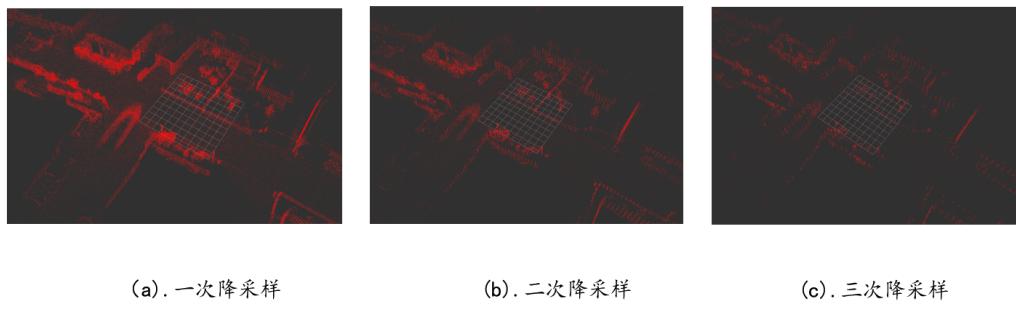


图 3.10 降采样结果效果图

laser scans for online operation” 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 提出的点云聚类方法，在一些细节处理上进行了创新，并将创新后改进的程序用于了底层导航系统设计。

1. 扫描线补偿

激光雷达的工作原理就是通过传感器发射激光光束并通过激光光束的返回时间差来实现对周围环境的感知，返回的形式最终以激光点类型记录在相应的文件中。如果传感器发射的激光雷达线束打在了反射率较低的物体上，会形成相应的点的缺失，例如车窗玻璃或者目前随处可见的玻璃落地窗等，都容易造成对当前环境的误判。这就造成了分析上的困扰。可以对得到的某一帧点云进行补缺，补缺的原理是通过线性插值的方式对确实的点进行补偿，方法如下。首先对无效点进行初步聚类，聚类得到的某一区域的无效点数量在阈值范围内，并且在阈值范围内的无效点周边的障碍物点距离不超过 1.5m，可以认为此处为确实的

障碍物点，通过线性插值的方式，对障碍物点内部包围的无效点进行补偿，补偿前的效果图和补偿后的效果图如下图所示。

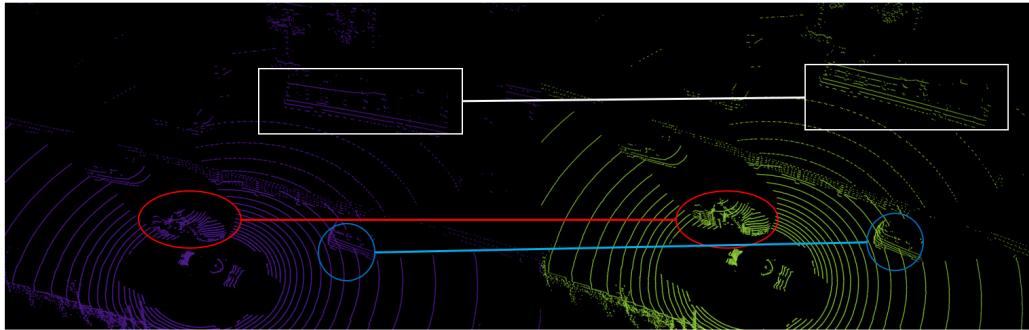


图 3.11 扫描线补偿前后对比

2. 地形分析

得到的局部点云是包含有静态物体、动态物体以及地面的综合帧，在处理数据的过程中，地面点通常来说是一个不许要过多考虑的因素，但是需要得到相应的地面点云，作为局部规划中的可行驶区域，并且得到地面点云之后才知道其他点相对于地面点的关系，得出此点是否是障碍物的结论。在 LeGOLOAM 中，点云中的地面点是经过实时分割的，分离出地面点与非地面点之后只保留非地面点部分进行数据处理。这样不仅可以降低计算量，也可以把去除地面点上一些干扰因素。去除地面点的方法有很多，第一步，首先考虑如何找到一帧点云的地面点。实验室的移动智能机器人上采用的是地面拟合法对点云中的地面点进行平面拟合。现实中激光雷达采集到的地面点通常不是一个完美的平面，那么采取单一的平面模型来拟合点云中的地面时不显示且具有相当大的噪点误差的。要完成相应的地面分割就需要对坡度进行一定的拟合，而不能够把坡度视为非地面点。通过将地面空间沿着 x 方向，分成若干个子平面，对每一个分割出的子平面进行地面平面拟合算法（GPF）进行拟合。GPF 的拟合算法如下所示：对于任意一帧点云 P ，最终的分割聚类的目的是将其分成 P_g 以及 P_{ng} ，即地面点和非地面点。算法在应用过程中，有四个十分重要的参数分别是 N_{iter} 、 N_{lpr} 、 TH_{seed} 、 TH_{dist} ，它们的意义是分别代表进行拟合迭代的次数、用于进行地面平面拟合的最低点代表（lowest point representative）的数量、选取种子点的阈值（点云中某点的高度低于最低点代表的高度和此阈值的和时，将此点加入种子点序列）、用于判断当前点云某点与拟合出的平面的正交投影距离的阈值（低于此阈值，则将此点加入 P_{ng} ）。

对输入得到的点云按照点云中每个点的 z 轴竖直进行排列，取 N_{lpr} 个最低

算法 3.1 算法示例 1

```

Data: this text
Result: how to write algorithm with LATEX2e
1 initialization;
2 while not at end of this document do
3   | read current;
4   | if understand then
5   |   | go to next section;
6   |   | current section becomes this one;
7   | else
8   |   | go back to the beginning of current section;
9   | end
10 end

```

点代表，并求得这些代表的均值 lpr_height 。如果某点的 p_z

$$p_z < lpr_height + TH_seed \quad (3.15)$$

那么便将此点加入种子点序列。确定平面模型时候，选择最基本的线性模型用于估计平面，线性模型

$$\begin{aligned} ax + by + cz + d &= 0 \\ \mathbf{n}^T \mathbf{x} &= -d \end{aligned} \quad (3.16)$$

这其实是一个公式，下式是对公式的向量化，其中 $\mathbf{n} = [a, b, c]^T, \mathbf{x} = [x, y, z]^T$ ，通过初始点集的协方差矩阵即可求得 \mathbf{n} ，从而确定出一个平面。采用种子点集为初始点集，通过公式

$$\mathbf{C} = \sum_{i=1:|S|} (s_i - \bar{s}) (s_i - \bar{s})^T \quad (3.17)$$

其中 \bar{s} 表示所有点的均值，其中协方差矩阵的三个向量可以通过奇异值分解(SVD)的方式求得，最终获得整个平面的法向量 \mathbf{n} 。上述得到这个初始的平面模型以后，会计算点云中每一个点到该平面的正交投影的距离，并且将这个距离与设定的阈值 TH_{dist} 比较，当高度差小于此阈值，认为该点属于地面，当高度差大于此阈值，则为非地面点。经过分类以后的所有地面点被当作下一次迭代的种子点集，迭代优化，在实际操作中选择的迭代次数为 3 次，既保证迭代出的平面拟合性，也可以保护计算资源。最终将拟合得到的去除地面点后的点云图以及地面点云分别如图所示：

得到了地面之后，根据拟合得到的地面对地面上的点与地面之间的高度进行分析，取其相对高度高于 15cm 的作为 Obstacle，即表示 obstacle 为障碍物点云，障碍物点云不可以直接通行，需要在避障模块中特殊考虑。高度在拟合地面与障碍物最低阈值之间的，视作斜坡，主要在局部导航的路径选择中作为权重之一，旨在选择出相同条件下，代价最低的局部路径。

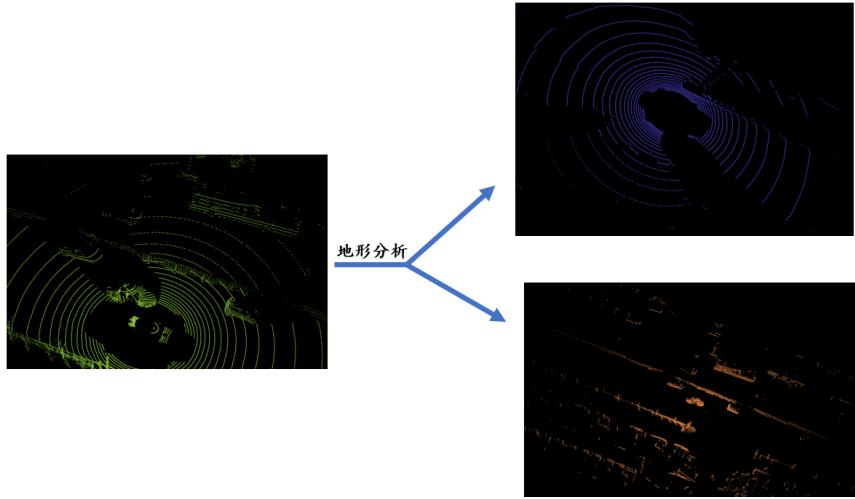


图 3.12 地形分析结果效果图

3. BFS 聚类方法

地形分析之后得到的地地面点云与非地面点云，地地面点云理所当然作为可行驶区域，不用做更加特殊化的处理。而对于非地面点云，则需要进行进一步的分割。首先对于非地面点云的聚类，进行四邻域搜索之后，设定阈值，将一定阈值内的点判断成为相同点簇，过程中可能会出现同一物体由于扫描角度等问题被分割成为两个点簇，故需要对两个点簇内的点进行角度判断，以确定两簇点云是否属于同一物体。如图，当两簇点云进行比对时，若任意两个点之间的角度 $\theta > \lambda$ ，则表示两点属于同一物体， λ 表示判断阈值。

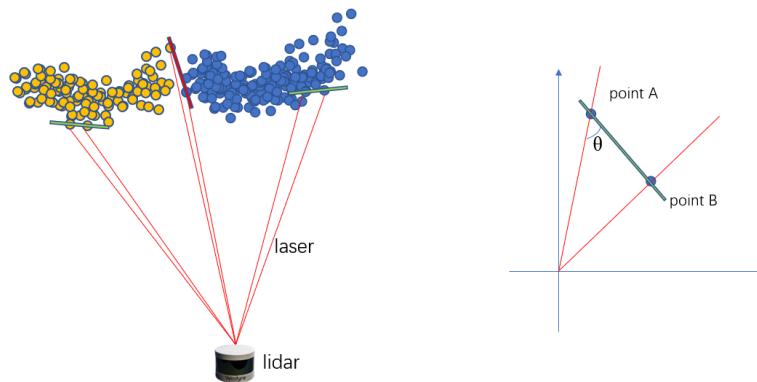


图 3.13 点簇融合与分割角度比较

3.3.3 底层局部规划器设计

本章上述几个模块已经对相应的环境信息进行了处理，并且可以拿到处理过后的环境点云信息。本节的具体内容是根据拿到的环境信息进行局部规划器的设计，规划器的设计参考了 CMU 团队的思路理念，融合实验室移动机器人的

阿克曼底盘模型，构建出了基于阿克曼模型的底层导航规划器。底层规划器的主要任务是处理相应的环境信息，订阅机器人在当前环境中的位置，将高层导航下发的次级目标点解算成为当前移动机器人所需要行走的方向与相应的速度。

1. 阿克曼底盘的轮式里程计模型

使用阿克曼底盘进行规划时，必须对阿克曼底盘的里程计模型进行了解，以进行正确的控制空间点采样。阿克曼的轮式里程计（以下简称轮式里程计）是根据车辆的运动学模型，将车辆的轨迹模型进行累计，得到车辆的运行里程的一种技术手段（以前此术语指的是一种仪器，现在多指一类技术方法，例如视觉里程计）。如图所示，现在有一个阿克曼结构的底盘模型示意图，如图。

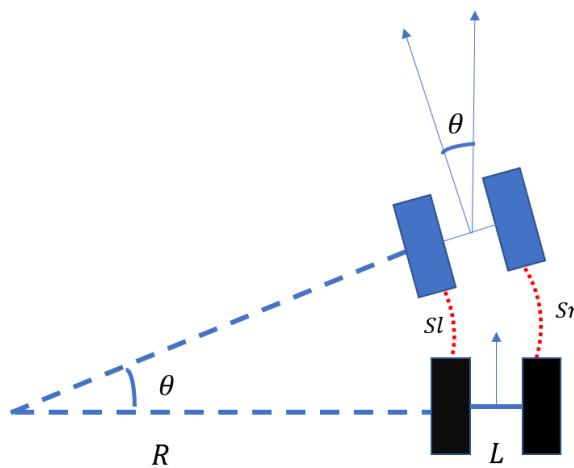


图 3.14 阿克曼结构示意图

阿克曼结构左右轮在转向时所走的距离不同： R 为转动半径， θ 为转动角度， L 为阿克曼轴距。其中：

$$\begin{cases} s_l = \theta R \\ s_r = \theta(R + L) \end{cases} \quad (3.18)$$

将两边的等式同时除以时间，进一步可以得到：

$$\begin{cases} v_l = \omega R \\ v_r = \omega(R + L) \end{cases} \quad (3.19)$$

最后再将两式联立可得：

$$\omega = (v_r - v_l)/L \quad (3.20)$$

证明阿克曼结构的角速度是相同的，更进一步，可以把阿克曼模型等效成为自行

车结构，对于移动机器人向前运动的情况， x 、 y 和 θ 三个维度上的速度分别为：

$$\begin{cases} \dot{x} = v \cos \theta \\ \dot{y} = v \sin \theta \\ \dot{\theta} = \frac{v \tan \theta}{L} \end{cases} \quad (3.21)$$

为了在计算机中表示和便于计算，我们将基于阿克曼底盘的运动学模型离散化为空间中的一系列点。这样，从初始时刻的初始位置出发，可以由前一时刻 t 的位置递归推导出 $t+1$ 时刻的位置和旋转角度。该方程表示为：

$$\begin{cases} x_{t+1} = x_t + v_t \cos(\theta_t) d_t \\ y_{t+1} = y_t + v_t \sin(\theta_t) d_t \\ \theta_{t+1} = \theta_t + \omega_t d_t \end{cases} \quad (3.22)$$

相应地，根据该离散公式可以得到对应的适用于阿克曼运动学约束的离散路径集。

2. 向前模拟三次样条生成离散路径组

在有了上面的阿克曼模型的里程计空间方程之后，以车辆的后轮中心为 `base_link`，以 `base_link` 为起点，按照阿克曼里程计模型，向前模拟三次，生成相应的离散路径组。离散路径组的示意图如图 3.5 所示，路径的转向范围是-22.5°~22.5°，完美贴合煜禾森 FR-07 的车身转向范围。第一次将 45° 转向范围分

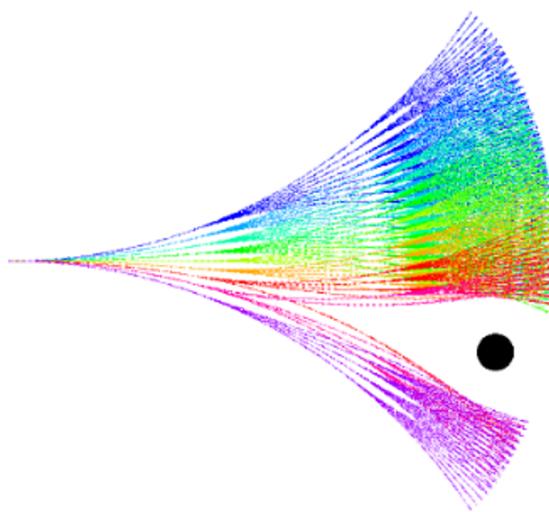


图 3.15 离线路径组示意图

为 9 个等分，每 5° 生成一条路径，然后按照此方式，在生成的 9 条路径的基础上二次模拟，直到三次模拟全部完成，共生成 9*9*9 条路径，一共分为 9 个 group，每个 group 种存有 9*9 条路径。

其中，图中黑色圆点表示有障碍物占据时，影响到的路径组数量。此处考虑了车身的本身具有的轮廓，路径的任何位置占据导致车身轮廓与障碍物点的碰撞，都会被视为影响路径。下文将详细叙述此部分。

将此路径组以一定的频率发送给底盘的后轮中心，并且需要注意的是，实际使用时候要将激光雷达扫射在移动机器人身上的部分，将这一部分的影响去除，因为车辆本身并不会撞到小车自身。在得到某一个目标点之后，局部规划期开始工作。

3. 依据点云信息判断路径上障碍物的方法

车身位置与路径组的关系如下图所示。路径组由车身后轮中心位置延伸出来，并且对路径组原点以棋盘中心，构建体素网格，体素网格与路径组内的点具有离线构建的对应关系。对应关系如下图所示：

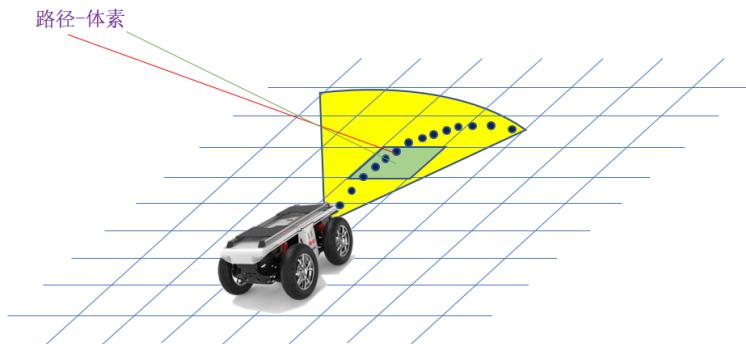


图 3.16 体素网格与路径之间对应关系示意图

当系统所处狭窄环境时，相应的可选择路径组及其系统的实际处境如图：本

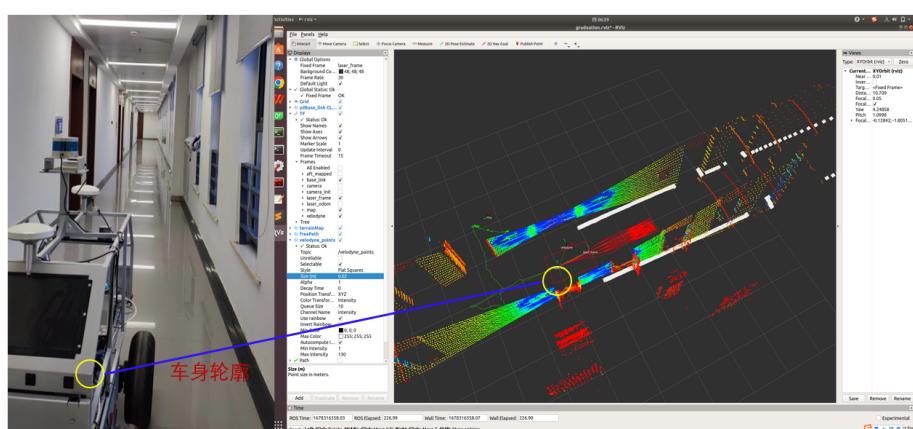


图 3.17 狹窄环境下的路径组与环境

文在处理实际的路径组和静态障碍物时，对车身进行了一定体积的膨胀，以车体的轮廓为基准，安全性为第一要求，对车身周围一定范围内的障碍物都保持一定的安全距离。

4. 最优路径选择

此刻，底层导航系统接收到一个局部目标点，位于车身的某个位置，如图所示

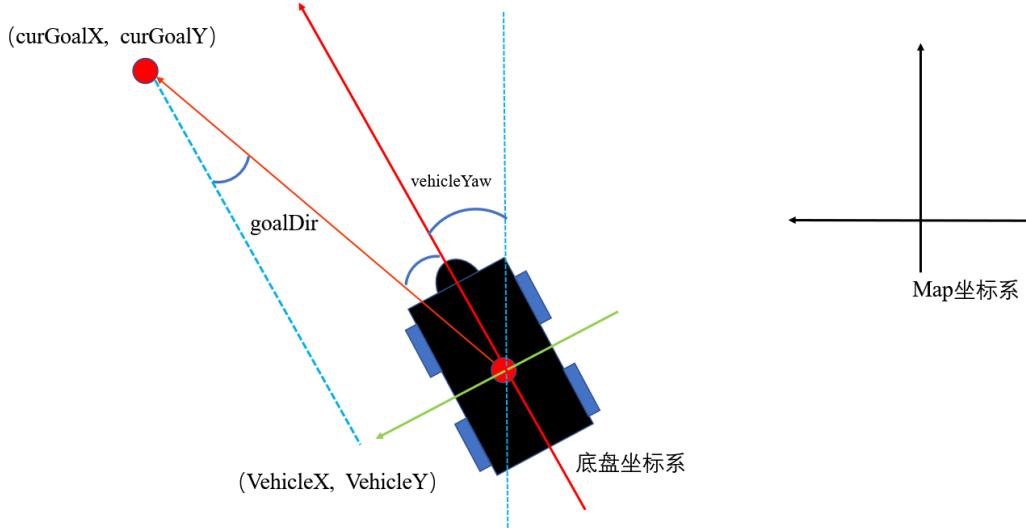


图 3.18 局部目标点与底盘朝向之间的角度拆解

图中所示的车身在全局坐标系下的朝向为 `vehicleYaw`，坐标为 `(VehicleX, VehicleY)`，局部目标点的全局坐标为 `(curGoalX, curGoalY)`，将目标点转换到底盘坐标系下，则目标点在底盘坐标系下的相对坐标为：

$$\begin{aligned} relativeX &= ((curGoalX - vehicleX) * \cos(vehicleYaw)) \\ &\quad + (curGoalY - vehicleY) * \sin(vehicleYaw)) \\ relativeY &= (-(curGoalX - vehicleX) * \sin(vehicleYaw)) \\ &\quad + (curGoalY - vehicleY) * \cos(vehicleYaw)) \end{aligned} \tag{3.23}$$

如图 5.6 所示为计算某条路径的速度方向的方式，在底盘坐标系下，要转向局部目标点的方向，需要的转向角度为 `goalDir`：

$$goalDir = atan2(relativeY, relativeX) \tag{3.24}$$

运动元路径组中共 $9*9*9$ 条路径中第 i 条路径的角度为 `pathDir`，那么第 i 条路径与目标点方向的角度差值为：

$$angDiff = |goalDir - pathDir| \tag{3.25}$$

对运动元路径组进行障碍物点云以及坡度点云按照式(3.16)方式转换到车体坐标系下，并且通过上述的体素网格与路径组的对应关系，剔除所有会影响到移动机器人车身的路径。使用 $\text{penaltyList}[i]$ 记录下路径中第 i 条路径上坡度点云的相对高度值，若没有坡度点云存在于第 i 条路径上， $\text{penaltyList}[i] = 0$ 。据此计算坡度惩罚项，记作 $\text{penaltyScore}[i]$ ：

$$\text{penaltyScore}[i] = 1 - \lambda * \text{penaltyList}[i] \quad (3.26)$$

λ 为系数，可以人为调节，影响移动机器人局部规划时对坡度点云的厌恶程度。

最后对第 i 条路径的评分为：

$$\text{score} = (1 - \sqrt{\sqrt{0.005 * \text{angDiff}}}) * \text{penaltyScore}[i] \quad (3.27)$$

如图 3.9 所示对每个 group 中的所有存在路径（不包含已经因障碍物遮挡剔除的路径）都进行遍历计算，最终得到整个组中所有的角度的评分和。然后排序所有的 group 评分，选择评分最高的那个 group 的路径组中的第一段，也就是三次模拟生成路径的第一段交给运动控制部分进行执行。

3.3.4 运动控制

采用路径跟随法进行运动控制，这也是本文提前生成离线运动元的原因之一。运动控制部分以选择好的路径组运动元为依托，每根运动元路径保存有相应速度信号。根据上述算法选择了路径组之后，相应的速度：linear 和 angular 都是确定的。

3.3.5 局部区域的底层导航方法验证实验

为了正式局部导航方法的鲁棒性与在实地场景使用的可行性，本文将按照局部导航方法构建的底层导航子系统以硬件形式搭建出来后，在校园的室内场景与室外场景下，分别进行了实验。为证实系统在室内室外场景下的功能通用性，在室内场景下，以 LeGO-LOAM 为底层导航的定位方式，在地下车库环境下进行路径点追踪实验；在室外场景下以 RTK 作为定位手段，通过预录制的路径点，围绕实验大楼做了一次底层导航的实验。实验的过程图片以及结果展示均来自于实地场景运行过程中的 ROS 系统封 rviz 截图，实验展示图中所有白线为发布的路径，蓝色线条为系统运行过程的实际路线实验结果以及过程中的终端界面以及定位方式的改变导致的 tf 树等变换关系呈现如下：

1. 室内基于 LeGO-LOAM 的局部导航实验

本文的主要工作在于通用系统的构建，因此对于定位的追求在于定位系统的室内外可切换性，并不过度追求定位模块的精度，且根据目前移动机器人系统

的计算核心算力考虑，采用计算资源消耗与定位精度平衡较好的 LeGO-LOAM 作为室内定位的方法。实验的真实场景与系统的界面如下图所示：

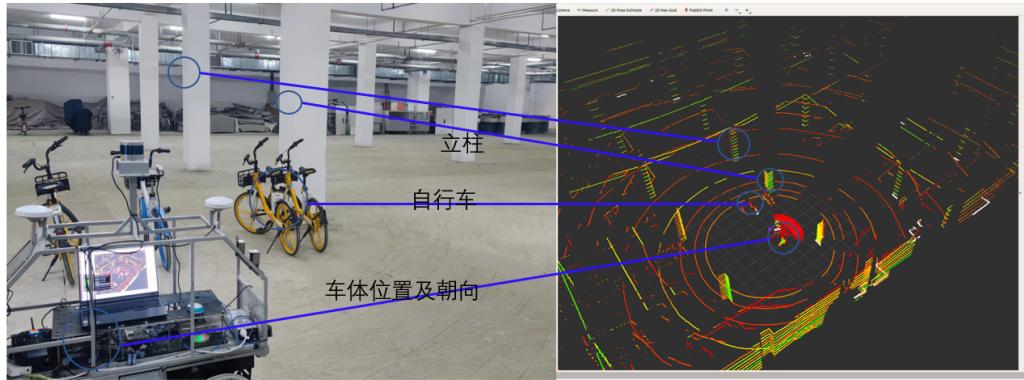


图 3.19 地下车库场景实物与系统局部点云对比

场景内有多处静态障碍物以及立柱，可以很好地显示出系统在无高层全局规划前提下，对室内目标点的追踪以及通过路径避障方式躲避静态障碍物的性能。

首先给系统发布一系列规则的点，模仿一定条件下对单点的导航能力。

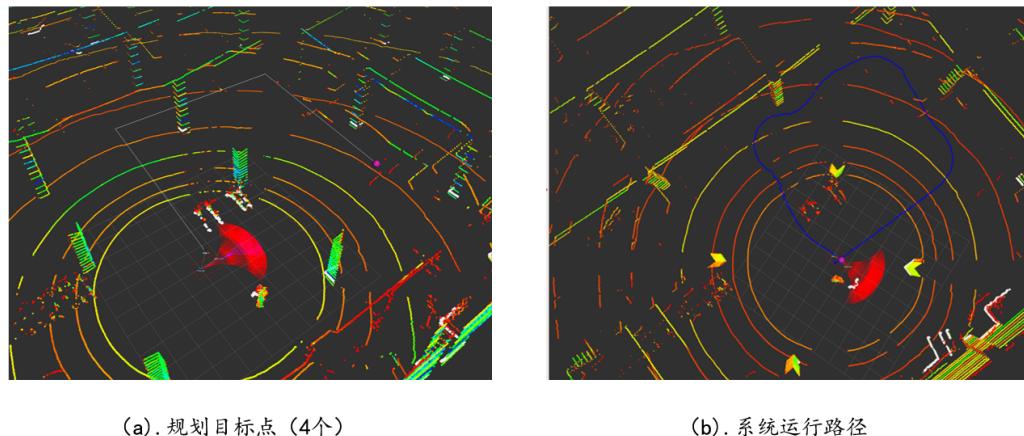


图 3.20 系统追踪指定多个目标点运行结果

上图（a）表示在系统运行之初规划的 4 个目标点的位置，四个目标点连成一个矩形，系统按照矩形运行一周，并最终回到初始位置。（b）中蓝色轨迹为小车运行时记录下的实际位置。可以看出系统可以按照目标点的给定鲁棒地追踪目标点。

然后按照上图目标点让系统二次运行，并在一定位置给予简单动态人体阻

碍，运行的结果和对比如下图所示：如图所示，系统的目标点不变情况下，在第

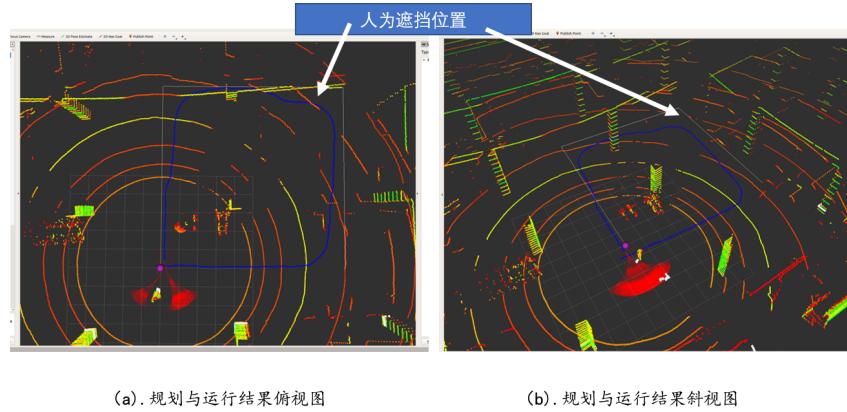


图 3.21 二次运行并施加动态遮挡

二个目标点处施加简单的动态遮挡，系统可以根据点云的位置作出实时反应，在判定到达目标点后向着下一个目标点做紧急转向，以避免可能发生的碰撞。同时系统二次运行的路线与第一次运行的轨迹有所差异，这是系统设计中并没有刻意去规定转向的优先级往左或者往右，而是给了系统在运行时更多的自主权力，为简单避障留有裕量。

下图将展示系统具体运行时的过程轨迹，展现完整的追踪目标点的过程，图中的紫色点是当前时刻系统所需要移动指向的目标点，在抵达某一个目标点后，才会向着下一个目标点进发，其详细过程和终端界面的输出如下图所示：

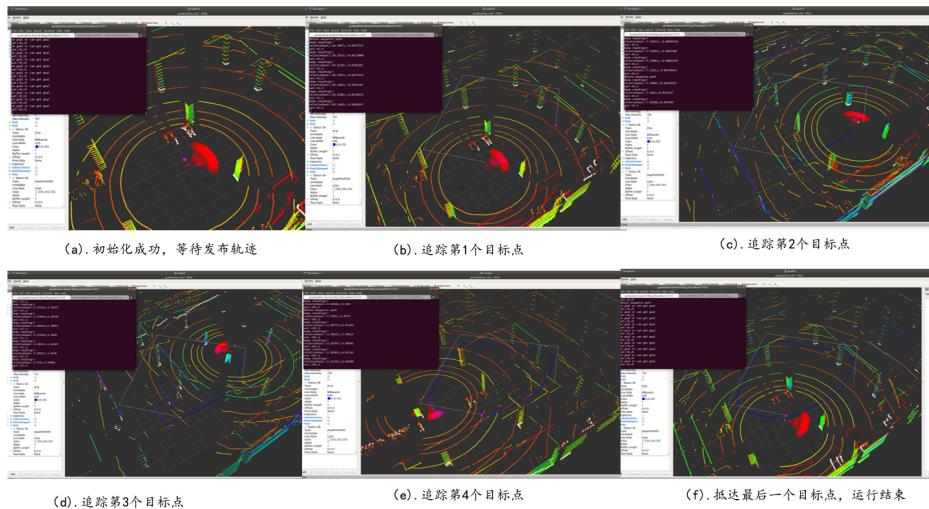


图 3.22 对目标点的完整追踪过程以及终端输出

如上图，在局部追踪线程就绪后，等待目标点发布后，会沿着一系列目标点行进，最后回到原定位置（也可以将终点目标定在非原始位置）。运行的过程图完整如上所见。

接下来为了展示系统在不同场景下的应用鲁棒性，在两个场景下，进行了路

点录制以及跟踪展示，录制路点并二次发布的过程是为了模拟在实际应用中高层导航下发的场景全局路径过程，同时，由于路点录制可以随心所欲，故可在录制过程中增加大量复杂性条件，可更好地测试底层导航的性能。

轨迹一：录制距离 3m，目标点的到达容忍度为 1.5m，结果展示如图：

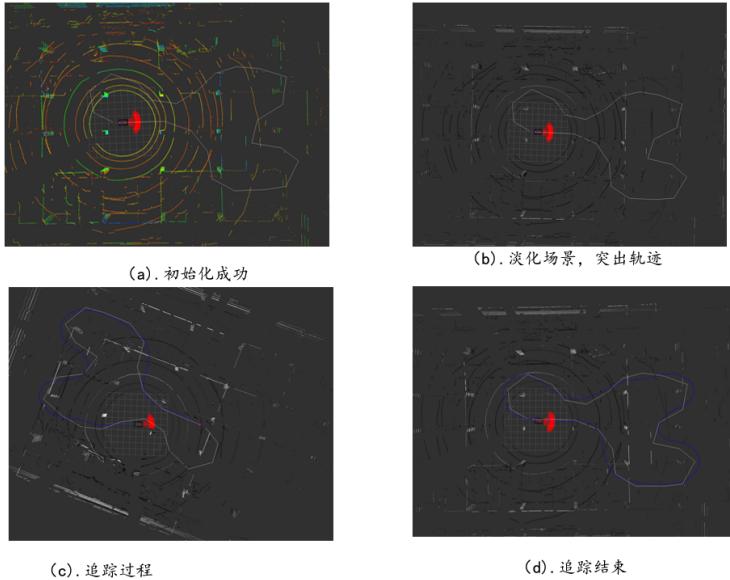


图 3.23 室内对路点的追踪过程

轨迹的对比结果如图所示：

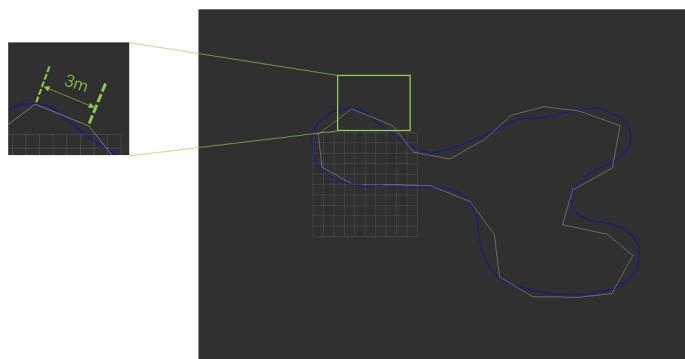


图 3.24 轨迹对比 1

此轨迹一录制共录制路点 27 个，全长约为 81m。录制路点的时候，设置路点之间最低距离为 3m，即白线折点之间的连线长度为 3m，车体到达目标点范围内 1.5 视为到达当前目标点，系统开始下发下一个目标点并追踪。

轨迹二：录制距离 1m，目标点的到达容忍度为 1m，图形为简单凸边形，结果展示如图：此轨迹一录制共录制路点 69 个，全长约为 69m。录制路点的时候，设置路点之间最低距离为 1m，即白线折点之间的连线长度为 3m，车体到达目标点范围内 1m 视为到达当前目标点。

轨迹三：录制距离 1m，目标点的到达容忍度为 1m，图形为复杂的轨迹曲线，

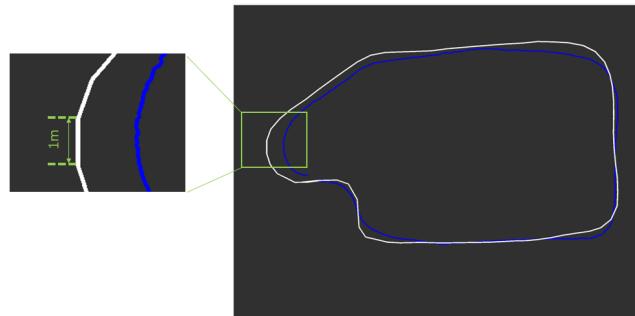


图 3.25 轨迹对比 2

结果展示如图：

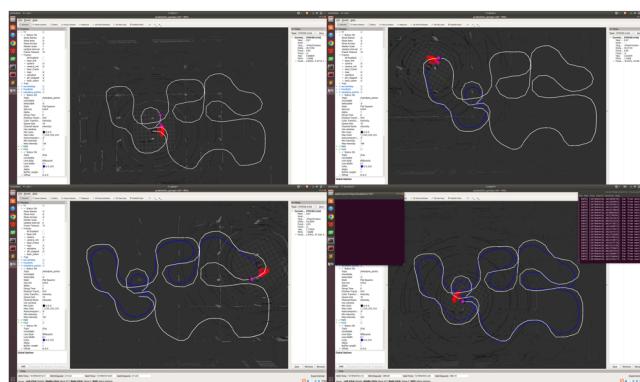


图 3.26 运行过程

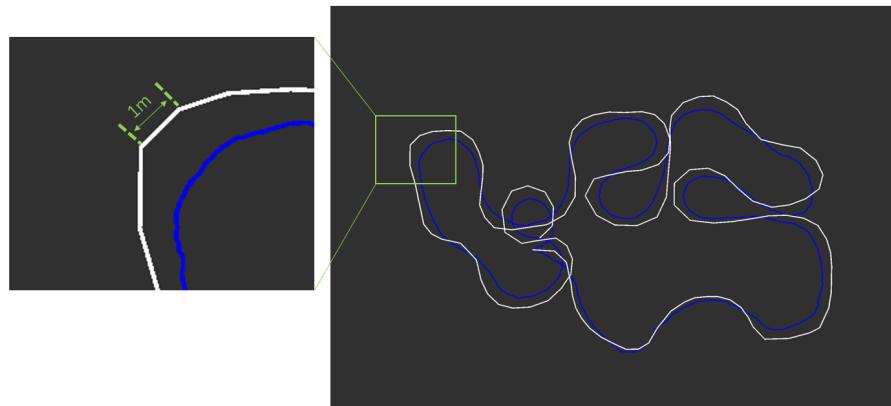


图 3.27 轨迹对比

此轨迹一录制共录制路点 140 个，全长约为 140m。

从以上几段轨迹的实验可以证明，我们搭建的移动机器人在室内可以使用 SLAM 方法对环境实时定位并跟踪相应的全局轨迹，随着算法的路点采集距离以及目标点的到达容忍度设置减小，轨迹跟踪的精度也愈高，但是受限于实验室阿克曼平台本身的体积和约束，目前的效果依然有待提升。若改换成体积较小的

移动机器人，算法的精准度可以再次提高。

2. 室外基于路点的大场景实验

上述实验的目的更多是证明相应室内导航算法的室内可用性，在室外使用RTK进行定位，并对在学校内的大型区域内进行了路点的录制与导航跟踪实验。本次实验的路点录制距离均为3m，到达目标的容忍度为2m。录制了两端轨迹，分别展示了园区运行的范围以及园区内运行过程中的多转弯等轨迹实验。室外的RTK选定的坐标原点以及车身在园区内对应的坐标系tf关系如图所示：

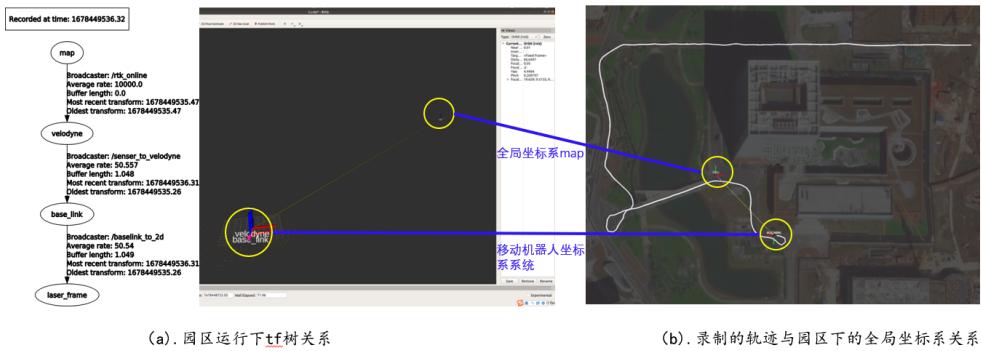


图 3.28 园区场景下的坐标系关系与实景对照 1

轨迹一：目标点的到达容忍度为2m，其过程展示图如下：

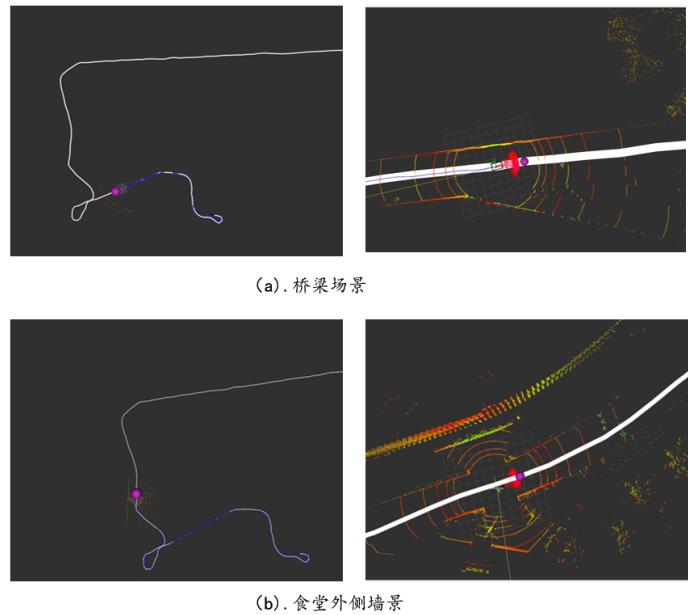


图 3.29 园区场景下的常轨迹追踪过程及其局部放大图

轨迹的对比结果如图所示：

此轨迹一录制共录制路点420个，全长约为1200m。

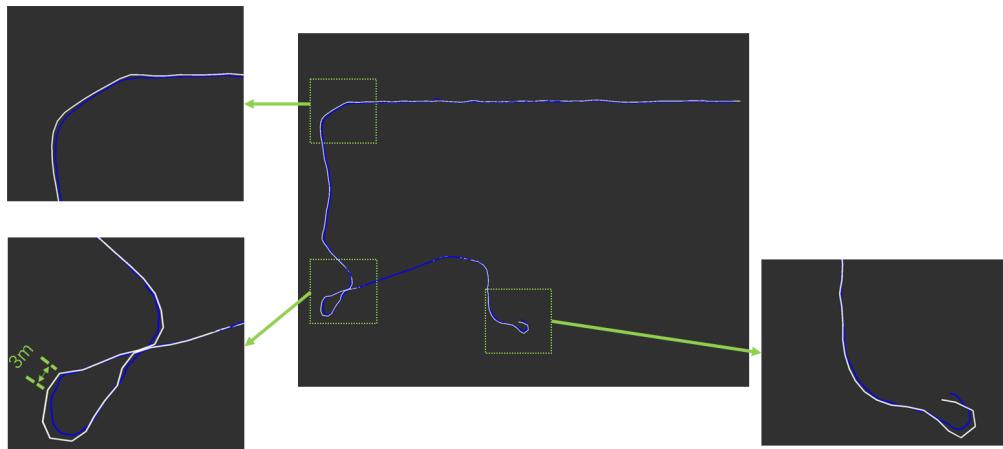


图 3.30 园区轨迹 1 对比及其局部放大效果图

轨迹二：目标点的到达容忍度仍为 2m，但选择的是园区内弯绕路线，测试系统在园区内的方向选择能力，实景图如下所示：

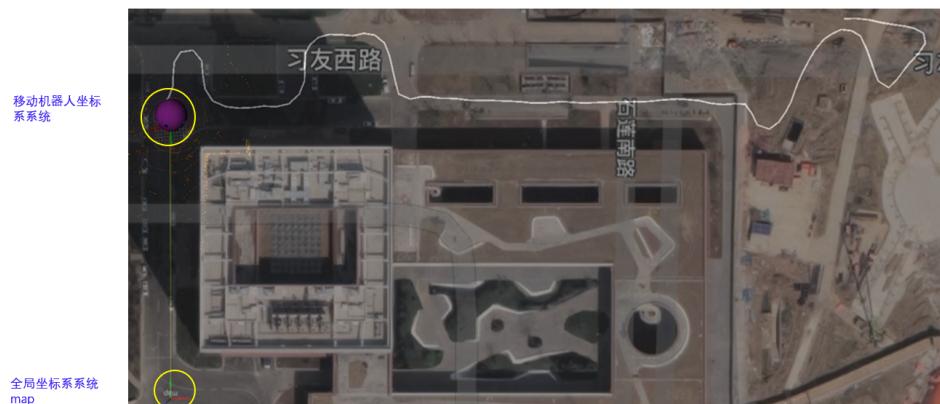


图 3.31 园区场景下的坐标系关系与实景对照 2

轨迹的对比结果如图所示：

此轨迹一录制共录制路点 130 个，全长约为 400m。

结论：经实验验证发现，在室外场景下，随着 RTK 的使用，其定位精度比室内场景下的 LeGO-LOAM 高，故导航系统的轨迹跟随准确性也随之升高。据此两个实验，验证底层局部导航的功能，并且验证了系统在室内外切换使用的可行性。

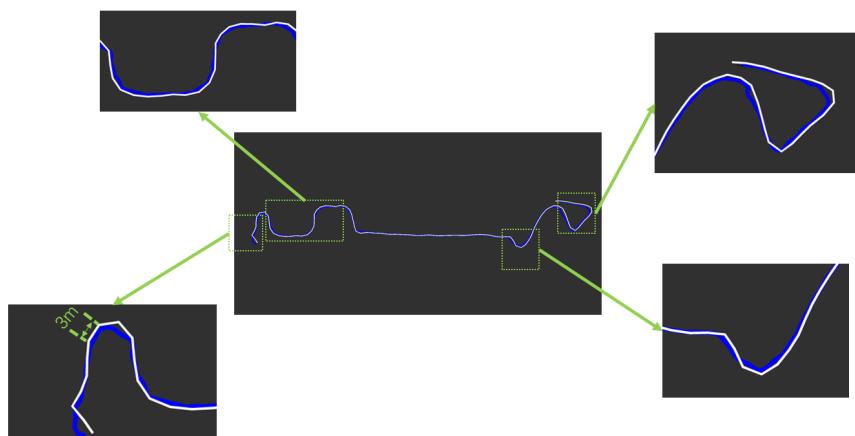


图 3.32 园区轨迹 2 对比及其局部放大效果图

第4章 基于ORCA的移动机器人导航避障系统

4.1 引言

智能无人车导航系统是移动平台完成给定任务的最基本支持模块。当智能无人车离开实验室的静态环境，步入园区环境后，与人或其他动态障碍物的接触是不可避免的。目前的导航避障思路十分明确且具有人类思考特征的：1. 在路径规划之后，对于确定的路线上的障碍物选择躲避；2. 对于即将到来的其他动态物体选择合适的路线提前反应，以避免因双方移动带来的可能性碰撞。但是目前的导航系统避障算法的研究多数都集中在第一层，即在自身行动路线明确的前提下，对路线上的某个时间段内要发生的碰撞进行反应。我们将这个反应称之为“应急避障”。已知的国内外多数机构乃至高校的移动机器人平台使用的多为此类算法，算法的最基础逻辑为碰撞避免，即安全性得到保证后，牺牲了部分的效率，并且可能陷入局部最优的问题。试想这样一个场景：移动机器人在园区内服务过程中，偶遇墙角或形势区域边界处有多位无序运动的移动障碍物，该怎么避免相应的碰撞呢？目前的主流的应对方法也有两种：1. 在导航的高层级上部署局部地图，对动态障碍物在地图上的位置进行实时更新以及预测，根据预测位置以及自身定位，规划处特定时间段内无碰撞的路径；2. 通过RVO、ORCA（也称RVO2），进行分布式的避障考虑。

为了解决实验室的智能无人车设备在校园场景下的问题，并结合工程情况实际，本文在ORCA思想原理下，并结合实验室智能无人平台本身的特性，研究设计出了一套适用于本平台的ORCA分布式导航避障算法。本算法不需要提前和其他的移动机器人进行通信，或者提前预测园区内行人等动态障碍物的轨迹，只需要进行简单观测便可使移动机器人自身进行轨迹修正，避免与其他动态障碍物发生碰撞。本章内容的介绍主要是在上述章节导航系统方法具备的条件下，为导航系统在园区场景内穿越动态区域提供算法保证。算法的具体流程如图3.1所示。

4.2 ORCA 算法基本原理

ORCA算法，英文全称是Optimal Reciprocal Collision Avoidance，最早在RVO的基础上演化而来，应用于时空交界下，移动式机器人相互碰撞避免场景。主要思想如下：一个任意移动物体的速度为 v ，包含有线速度linear和角速度angular，若将可能会发生碰撞的速度矢量看作一个集合，称之为 VO （Velocity Obstacle）。如果把所有动态障碍物方向的速度排除在集合以外，可以得到关于无碰撞速度



图 4.1 模板图片

的集合，那么移动机器人在无碰撞的速度集合中取到的速度都不会发生碰撞，问题在于如何确定这个速度集合。本章在讨论动态避障问题时候，不对行人动态行为做深层次预测，将行人与其他动态障碍物做模式等效，也就是通过对当前速度的观测，假定在一段时间内，行人的运动模式不发生变化（实际上，园区内的行人运动方式的确有规律可循），同时本文讨论的避障方法应用于多层级导航系统中，系统本身带有的避障算法可弥补因统一模型而带来的可能碰撞缺陷，因此本文讨论的前提是将行人与移动机器人的运动模式等效，且确认此种等效在本导航系统下不会出现系统性风险。

4.2.1 ORCA 在导航中应用的问题描述

当移动机器人在环境中导航运动时，可行驶区域内一定范围出现另一动态障碍物，此时可虑让这二者不发生碰撞，将移动机器人和此障碍物归一化考虑，使用同一种模型去考虑不同的动态障碍物类型，以下统称为动态障碍物。对于任意的两个动态障碍物 A 和 B，速度障碍 $VO_{A|B}^{\tau}$ 表示 A 相对于 B 的所有相对速度的集合，这个速度会导致 A、B 两个动态障碍物在 τ 时刻内发生碰撞。

为描述的方便性以及公式引出的合理性，将导航过程中所有可能出现的动态障碍物都等效成圆模型，以便对过程的推到。可定义为：

$$D(p, r) = \{q | \|q - p\| < r\} \quad (4.1)$$

$D(p, r)$ 表示圆心位置在 p 处的，半径为 r 的圆模型， q 表示圆模型内的任意一点。

圆模型及 A、B 关于原点的对称圆模型如下所示：

据此可以用数学语言描述 $VO_{A|B}^{\tau}$ ：

$$VO_{A|B}^{\tau} = \{v | \exists t \in [0, \tau] : tv \in D(p_B - p_A, r_A + r_B)\} \quad (4.2)$$

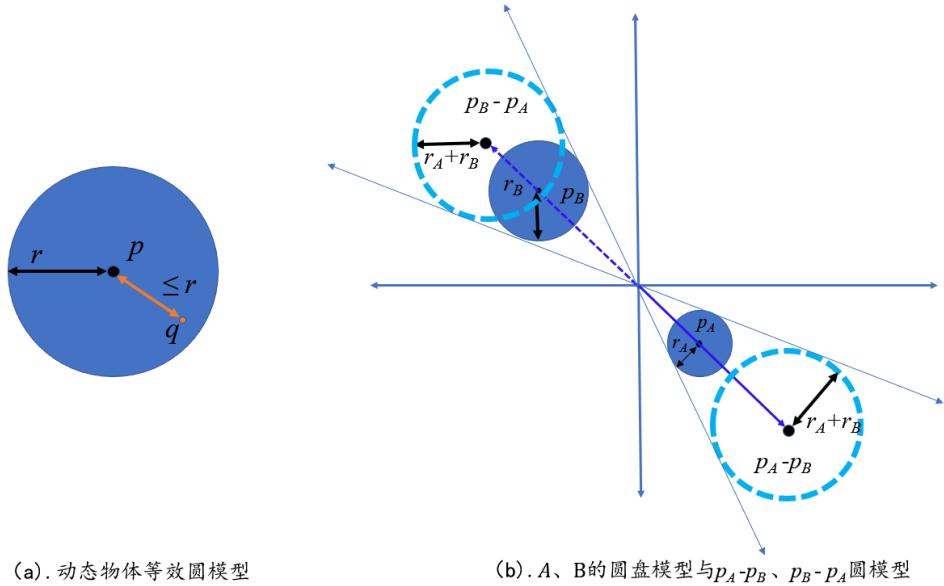
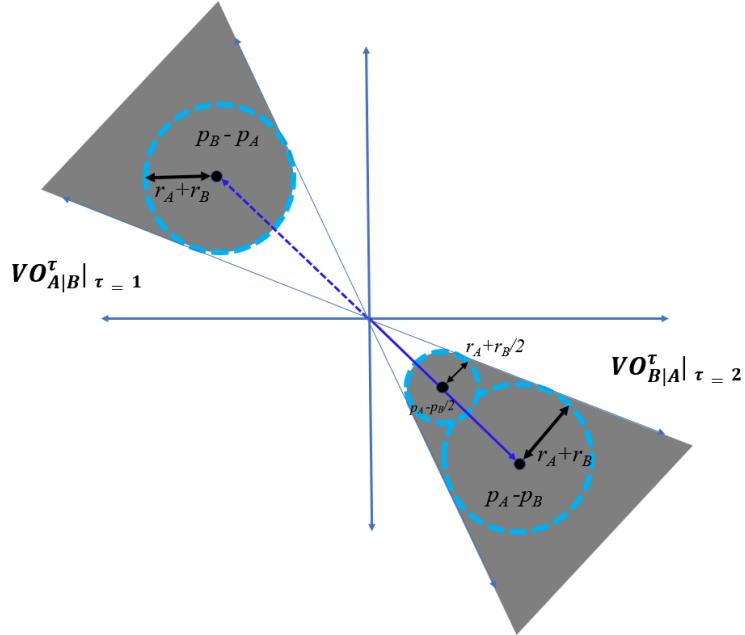


图 4.2 圆模型以及 VO 的分析图

速度障碍可以由几何表示，如图： $VO_{A|B}^\tau$ 与 $VO_{B|A}^\tau$ 在 τ 相同时是关于原点对称的截断锥形，如图 4.3 所示。


 图 4.3 $VO_{A|B}^\tau$ 与 $VO_{B|A}^\tau$

对称的截断锥形，从图 4.2 (b) 可知 $D(p_B - p_A, r_A + r_B)$ 与 $D(p_A - p_B, r_A + r_B)$ 也是关于原点对称的。对上图中 $VO_{B|A}^\tau | \tau=2$ 可用物理描述解释为：若 B 相对于 A 的速度处于图 4.3 的灰色区域之中，那么在时间窗口 $\tau=2$ 之前的某个时刻，A、B 之间将发生碰撞。以上是速度障碍的最直观解释。

由此特例推广开来，对于 A、B 两个物体的速度 v_A 、 v_B ，如果 A、B 在 τ 时刻内继续以当前速度运动，那么，A、B 必将在某个时刻发生碰撞。此过程可以

等效描述为：

$$\begin{aligned} \mathbf{v}_A - \mathbf{v}_B &\in VO_{A|B}^\tau \\ \text{or :} \end{aligned} \quad (4.3)$$

$$\mathbf{v}_B - \mathbf{v}_A \in VO_{B|A}^\tau$$

如果从这个角度出发，要让 A、B 在一定时间内不发生互相碰撞，可以通过： $\mathbf{v}_A - \mathbf{v}_B \notin VO_{A|B}^\tau$ 来完成。那么可以至少保证在 τ 时刻内，A、B 两个物体互相之间无碰撞。由此可得，当 B 在集合 V_B 中选择速度 \mathbf{v}_B ，即 $\mathbf{v}_B \in V_B$ ，对于 A 物体来说，若要两者不发生碰撞，相当于

$$\mathbf{v}_A \notin VO_{A|B}^\tau \oplus \mathbf{v}_B \quad (4.4)$$

式中 \oplus 表示明可夫斯基和，即集合的并集关系。由 \mathbf{v}_A 的取值范围，可以引出 A 物体相对于 B 的避碰速度集（Collision Avoidance）：

$$CA_{A|B}^\tau(V_B) = \{\mathbf{v} | \mathbf{v} \notin VO_{A|B}^\tau \oplus \mathbf{v}_B\} \quad (4.5)$$

也即 $\mathbf{v}_A \in CA_{A|B}^\tau(V_B)$ 时，A、B 在 τ 时间内不会发生碰撞。

$CA_{A|B}^\tau(V_B)$ 可用几何图形解释为如图：图 4 中所有灰色之外均为避碰速度。

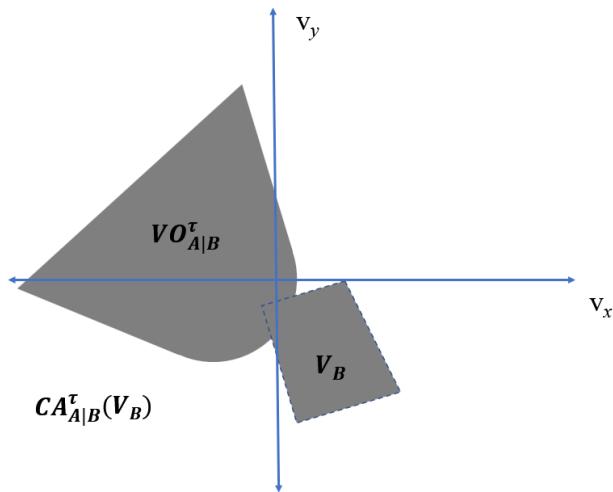


图 4.4 A 相对于 B 的避碰速度集几何解释

定义集合为属于 A、B 物体的避碰速度集合的子集，当两个子集分别和其父集合重合时，此时的 V_A 、 V_B 称为最大速度避碰集合。

此时如果有两个任意速度集合 V_A 、 V_B ，且这两个速度集合又是 $CA_{A|B}^\tau(V_B)$ 、 $CA_{B|A}^\tau(V_A)$ 的子集，那么称这两个集合 RCA (reciprocally collision- avoiding) 集。基于上述叙述，在选择避碰速度集合时，希望为 A、B 直接在 $CA_{A|B}^\tau(V_B)$ 、 $CA_{B|A}^\tau(V_A)$ 中选择速度，以保证 A、B 在时间窗口 τ 内不会发生碰撞并且可选择的速度最多。

在本文搭建的导航系统下，移动机器人在接受了目标点之后，会有一个最优的速度接近目标点，那么按照上述推理，分别将机器人与人都视作圆盘模型，则希望找到一个集合最大限度地“接近”导航系统当前时刻需要地最优化速度 v^{opt} ，将这个集合定义为 ORCA 集合。若两个物体分别是仍记作 A、B，ORCA 则记作 $ORCA_{A|B}^\tau$ 、 $ORCA_{B|A}^\tau$ 。

4.2.2 ORCA 的求解

求解 ORCA 的过程不需要通过大量的非线性计算，而可以通过集合构造求解的方式进行，如图：在此刻假设移动机器人分别采用最优化速度 v_A^{opt} 、 v_B^{opt} ，且这



图 4.5 模板图片

会导致 A、B 之间产生碰撞。即：

$$\mathbf{v}_A^{opt} - \mathbf{v}_B^{opt} \in VO_{A|B}^\tau \quad (4.6)$$

而 \mathbf{u} 为 $\mathbf{v}_A^{opt} - \mathbf{v}_B^{opt}$ 到速度障碍边界上最近的点的向量，则 \mathbf{u} 是 A、B 在时间窗口 τ 内避免碰撞所做出的相对速度的最小变化，即：

$$\mathbf{u} = (\underset{\mathbf{v} \in \partial VO_{A|B}^\tau}{\arg \min} ||\mathbf{v} - (\mathbf{v}_A^{opt} - \mathbf{v}_B^{opt})||) - (\mathbf{v}_A^{opt} - \mathbf{v}_B^{opt}) \quad (4.7)$$

\mathbf{n} 为速度边界处在点 $\mathbf{v}_A^{opt} - \mathbf{v}_B^{opt} + \mathbf{u}$ 的法线。则 \mathbf{u} 是 A、B 在时间窗口 τ 做出碰撞避免所需的最小相对速度变化。一般情况下，在多机器人系统的避障策略中，以对半分摊最小相对速度变化的方式计算 ORCA，即：

$$ORCA_{A|B}^\tau = \{\mathbf{v} | (\mathbf{v} - (\mathbf{v}_A^{opt} + \frac{1}{2}\mathbf{u})) \cdot \mathbf{n} \geq 0\} \quad (4.8)$$

在本导航系统的考虑中，为了安全准则考虑，假设行人、其他车辆等障碍物对速度的调节量降低，不以常用的对半分摊最小相对速度变化的方式考量，那么

对于选定的代理机器人而言：

$$ORCA_{A|B}^{\tau} = \{v | (v - (v_A^{opt} + \frac{3}{4}u)) \cdot n \geq 0\} \quad (4.9)$$

求解图中的 u , 有表达式：

$$u = (\arg \min_{v \in \partial VO_{A|B}^{\tau}} ||v - (v_A^{opt} - v_B^{opt})||) - (v_A^{opt} - v_B^{opt}) \quad (4.10)$$

可以解释为假设 A 和 B 的自适应速度分别为 v_A^{opt} 、 v_B^{opt} , 那么 A、B 会碰撞, 则 $v_A^{opt} - v_B^{opt} \in VO_{A|B}^{\tau}$, u 是以 $v_A^{opt} - v_B^{opt}$ 为起点, 指向最接近 $ORCA_{A|B}^{\tau}$ 边界的点的向量。

求解图中的 n , 有表达式：

$$n = (v_A^{opt} - v_B^{opt}) + u \quad (4.11)$$

表达式可以解释为 n 是以 $VO_{A|B}^{\tau}$ 的边界上的点 $(v_A^{opt} - v_B^{opt}) + u$ 为起点, 向外延伸的法线。

求解 $ORCA_{A|B}^{\tau}$, 有表达式：

$$ORCA_{A|B}^{\tau} = \{v | (v - (v_A^{opt} + \frac{1}{2}u)) \cdot n \geq 0\} \quad (4.12)$$

此关键表达式被解释为: $ORCA_{A|B}^{\tau}$ 是一条被直线分割开的一个半平面集合, 这个半平面集合为 n 指向的一侧。其中这条直线垂直于 u , 且穿过点 $v_A^{opt} + \frac{1}{2}u$, 由两点确定一条直线。

4.2.3 多个机器人碰撞的情况

上述情况针对于仅有两个移动机器人的碰撞情况, 当实际情况中出现多个机器人时, 需要在确定了一个主题之后, 对其他所有的移动机器人归为一类, 都视为当前机器人 A 的 B。即对于机器人 A 的所有 B 有

$$ORCA_A^{\tau} = D(0, v_A^{max}) \cap \left(\bigcap_{A \neq B} ORCA_{A|B}^{\tau} \right) \quad (4.13)$$

也就是说此时 $ORCA_A^{\tau}$ 是机器人 A 相对于其他任意机器人 B, 以它的最大速度 v_A^{max} 为半径的圆与其他所有 B 的 ORCA 的交集。而后机器人在集合允许的速度范围内选择最接近其设定的最优速度 v_A^{pref} 的新速度 v_A^{new} :

$$v_A^{new} = \arg \min_{v \in VO_A^{\tau}} ||v - v_A^{pref}|| \quad (4.14)$$

此时求得的速度便是允许的速度集合中最接近最优速度的速度值, 机器人的位置便可以旧的位置 p_A 加上速度乘以时间求得。

4.3 基于ORCA的系统级避障策略

第3章对系统方法描述中介绍了系统运行过程中使用点云点检测和体素网格方式进行近距离范围的避障策略。此避障方法简单而有效，在静态场景以及很少动态物体出现的区域时候表现良好，但应对多动态区域时会使得整个规划模块混乱。因此在穿过多动态区域时，需要对此方法进行改良，并且与ORCA的线程结合。两者结合的有点至少有两个：1. 保证移动机器人穿过多动态区域时具有优秀的规划与避障能力；2. 将ORCA原本的静态障碍物规划过程免去，只需要计算对应的动态障碍物速度障碍即可，减少了系统的计算量。

4.3.1 多动态区域的移动物体观测与速度计算

要开展ORCA的计算，首先需要对考虑范围内的移动障碍物进行速度观测或估计。本文第三章3.3节已经详细介绍了对周边障碍物的分割与聚类过程，可以得到需要观测环境内的动态障碍物的实例分割。接下来对相应的实例进行简单地速度观测。

此处借助高层导航构建地先验栅格地图，此地图以园区内图书馆某处为坐标系原点建立，故可以用以下示意图表示动态障碍物的占据与移动趋势。本文进行ORCA避障的前提不对实例中每个点进行分析，而是对占据的栅格进行动态占据的方式分析被占据栅格的移动趋势。移动机器人在全局地图上的移动仅限于可行驶区域，故首先提取可行驶区域上的动态障碍物点云信息。

上述得到分割后的实例之后，确定可行驶区域上的动态障碍物方法是通过高度阈值进行判断，判断的方法基于动态物体（例如车、行人等）与地面有接触。上一章中已经对地面点和可行驶区域以及移动机器人周围的实例对象进行了分割介绍。本文此部分更加着重于对动态障碍物的分辨和移动趋势计算。将此刻移动机器人附近的可行驶区域与之前高层导航方法中分割出的地面点进行对比，即可得到此时可行驶区域上新出现的障碍物点云，由于园区场景内静态障碍物位置与区域基本不发生变化，故新多出的障碍物点云即可认为是移动障碍物。对移动障碍物点云的最高点高度值 max_height 记录并且记录此刻点云在栅格地图上所占据的栅格索引值 index ，设置点云的标签为：

$$\text{label} = \alpha * \text{max_height} + \beta * \text{index} \quad (4.15)$$

其中 label 是对点云的记录标签， α 、 γ 分别代表点云高度最高值和索引值在标签中所占的比例系数。在高动态场景下， α 所占比例应当适当调高，将关注重点放在动态物体的本身特质上。对场景内可行驶区域上的所有动态点云建立标签集，当下一次扫描来临时，对扫描得到标签集进行阈值搜索，匹配两次扫描的标签集。认为当前扫描的标签集是动态障碍物两次扫描的对应位置变化。接下来

计算相应物体的移动速度。

示意图如下所示：

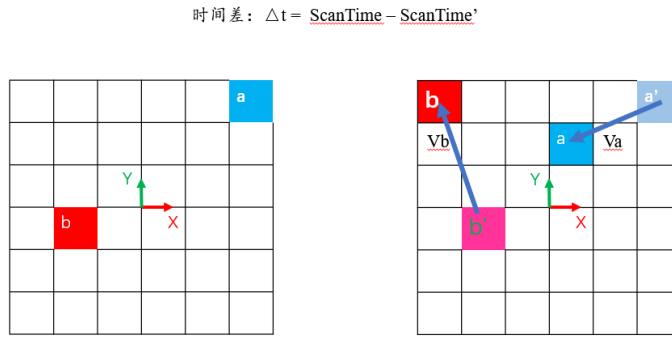


图 4.6 全局栅格地图与动态障碍物的占据与移动趋势

移动障碍物在全局栅格地图上的占据位置随时间发生变化，任意一个移动障碍物的速度可以估算为：

$$V = \frac{\sqrt{(Voxela'.y - Voxela.y)^2 + (Voxela'.x - Voxela.x)^2}}{\Delta t} \quad (4.16)$$

`Voxela'.y`、`Voxela'.x`、`Voxela.y`、`Voxela.x` 分别表示扫描时刻 `ScanTime'` 和 `ScanTime` 时刻被占据方格中心的坐标值； Δt 是两次扫描的时间间隔。速度的方向即为图中箭头所示的方向，图中的坐标系指的是校园中选定的全局坐标系的位置。至此，移动机器人周围的动态物体的移动速度即可通过观测得到。可以开始下一步的速度障碍的计算。

4.3.2 ORCA 与系统的结合方案

在 ORCA 模块不启动情况下，移动机器人的导航系统会根据当前的目标点在局部规划时给出相应的路径，即运动元路径组中的某一条。ORCA 避障模块启动后，会将当前路径组中选定的路径的方向和速度大小作为参考值即前面原理中提到的 V_{pref} 。离线运动元路径组是以点的形式存于 txt 文件中，并在每次程序运行时启动并读取存于相应的 vector 容器中。由于离线路径组的转向角度并不同于路径组的路径方向，故需要对路径组的路径速度方向计算。离线路径组的速度方向的计算方式如下图所示：

图中运动元在路径终点处的点为 P，坐标值分别为 a、b。那么移动机器人选择此条路径的移动速度方向即为：

$$V_\theta = atan2(b, a) * 180/\pi \quad (4.17)$$

此速度方向即为 ORCA 计算速度原理中 V_{pref} 的速度方向，大小则直接读取相应

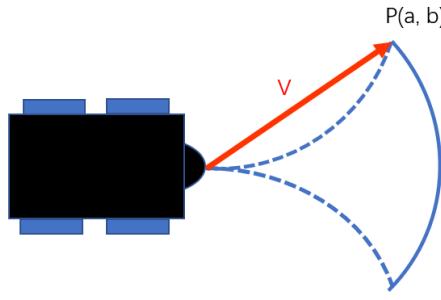


图4.7 离线运动元路径组的速度方向计算

的路径对应的默认速度大小。

然后根据上述的观测，确定一定范围内的可行驶区域中存在的动态障碍物数量，据此通过ORCA库中的Agent::computeNewVelocity()函数计算出上述计算原理中的 v_{new} ，原始的ORCA算法在此处即可完成计算，并且将 v_{new} 输入给机器人系统执行运动。但此处计算出的速度是将机器人当作圆盘模型构建的，拥有 360° 无死角的全方位原地转向能力，故此速度在实际移动机器人底盘上无法执行。本文提出的算法在速度计算之初将模型当作圆模型进行处理，也就是说，得到的速度 v_{new} 是阿克曼模型中心位置所需要进行运动的速度和方向，但是很明显，阿克曼模型中心并不能随心所欲地按照 v_{new} 给出的速度大小和方向运动，因此必须得在得到计算结果后对速度 v_{new} 与阿克曼的模型进行适配。计算得到 v_{new} 之后做出下处理：

- (1) .若检测到移动机器人的位置已经在终点处，则令速度为0，停止规划；
- (2) .计算当前移动机器人的朝向yaw角度与 v_{new} 之间的角度差值 $\Delta\theta$ ，若速度角度差 $\Delta\theta$ 在阈值之外，也就是说移动机器人不可以通过走直线来接近局部目标点，同时表示移动机器人在目前朝向下不可能向着局部目标点方向出发，于是低速调整速度方向：取 $\Delta\theta$ 符号sign，使用阿克曼模型的最大转向角度向着偏差方向慢速调整速度方向；
- (3) .若 $\Delta\theta$ 在转向阈值之内，表示机器人运动元路径组的覆盖方向包含有 v_{new} 的方向，并且此时移动机器人没有到达目标点，需要将此速度 v_{new} 适配到阿克曼底盘上，因ORCA计算过程是将速度转化成向量及其投影简化计算，其计算出的 v_{new} 以向量形式表示，在x、y方向上的速度分量分别是($v_{new.x}, v_{new.y}$)，不能直接使用，需要转化成线速度（推进速度）以及角速度的形式，才可成为阿克曼底盘可用的速度：

必须明确的是在本实验平台下，阿克曼底盘移动机器人按照 v_{new} 进行移动，指的是移动机器人在路径组的group中的第一段路径终点位置与路径起点的连

线方向与 v_{new} 一致。

首先需要确定移动机器人底盘的角速度：当前时刻移动机器人在全局坐标系下的朝向为 θ ，且当前所需要的速度方向为 $\text{atan}2(v_{new}.y, v_{new}.x)$ ，根据上述介绍的离线路径组的速度方向，选择与 v_{new} 速度方向最接近的一条路径，通过此路径对应的索引查询到此路径本身带有的角速度信息 $angular$ 。

确定要走的路径之后，接下来确定线速度 v_{linear} 的大小：在任意时刻，沿着选定的运动元路径行驶时，运动机器人的当前线速度为 v_{linear} ，移动机器人的朝向与 v_{new} 的夹角为 φ ，如图所示：有等式：

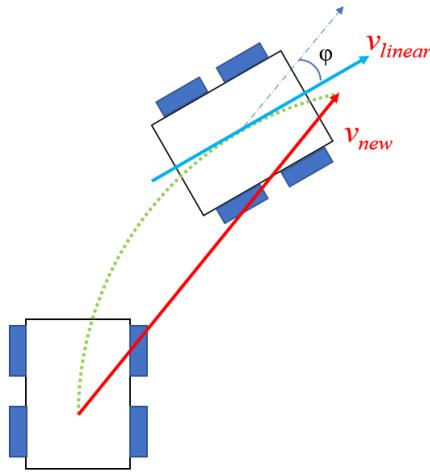


图 4.8 阿克曼模型的速度方向与大小等效方式

$$v_{linear} = \sqrt{(v_{new}.x)^2 + (v_{new}.y)^2} \cos \varphi^{-1} \quad (4.18)$$

v_{linear} 即为发送给阿克曼底盘的线速度（推进速度）。

上述过程最终可以得到相应的最终速度， v_{linear} 和 $angular$ 这两者组合成的速度便可以输入回规划系统并传给运动控制模块进行机器人运动控制。

4.4 动态避障系统的仿真实验

实验环境：Ubuntu 操作系统，ROS 机器人操作系统，构建阿克曼运动学模型于 rviz 中，并通过 stage_ros 内部传递多个机器人之间的速度与位置信息。

实验设置：该实验通过 rviz 与 stage_ros 作为实验环境，通过 xarco 文件构建阿克曼机器人模型，并将模型置于 rviz 界面中，通过 stage_ros 作为信息状态的处理中心，机器人状态的共享是通过使用 ROS 中的机器人状态发布器 (robot_state_publisher) 和关节状态发布器 (joint_state_publisher) 来实现的。

首先，在每个机器人的命名空间中，启动一个 robot_state_publisher 节点和一

个 joint_state_publisher 节点。robot_state_publisher 节点订阅机器人的 URDF 模型和机器人的关节状态，然后将机器人状态发布到 /tf 主题上。joint_state_publisher 节点从 /joint_states 主题订阅关节状态，并将其转换为 /tf 主题上的机器人状态。

然后，在每个机器人的命名空间中，启动一个 move_server 节点和一个 move_client 节点。move_server 节点负责处理机器人的运动控制，例如机器人的速度控制和路径规划。move_client 节点是一个简单的命令行工具，用于向 move_server 发送命令控制机器人运动。

在最后一部分的 static_transform_publisher 节点中，发布一个静态的 TF 变换，将机器人的 /map 坐标系和 /odom 坐标系对应起来。这样就可以在 rviz 中显示机器人在地图上的位置。由于机器人状态发布器和静态 TF 发布器发布的是相同的机器人状态，因此在 rviz 中看到的机器人位置和在 Stage_ros 中看到的机器人位置是相同的，也可以通过改变 stage_ros 中机器人的状态来影响 rviz 中机器人的状态。

实验中所使用到的栅格地图环境使用“map_sever”服务器进行加载，主要加载的是无障碍物的栅格地图区域，目的是执行 ORCA 的动态避障效果。此处实验不考虑静态障碍物，此部分已经在第 3 章中详细叙述并且实验证明。

rviz 下建立的阿克曼机器人模型如下图所示：

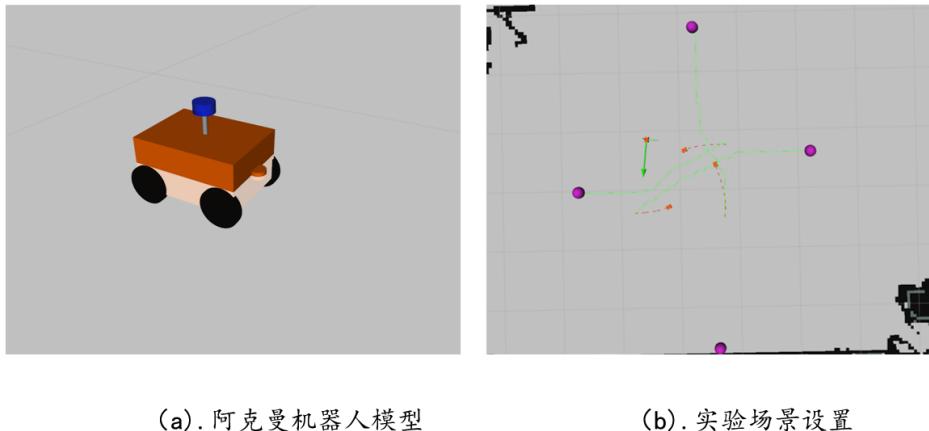


图 4.9 仿真环境中的阿克曼模型和场景

阿克曼模型依据实验室移动机器人的外形作为设计参考，具备底盘、三维激光雷达和 2D 激光雷达。场景中紫色球部分为实验为移动过程中移动机器人需要到达的指定目标点，通过指定目标点的形式模仿实际应用中不同动态物体向着不同方向和趋势移动的过程。绿色实线是在栅格地图上使用 A* 算法规划处的路径，目的是按照移动机器人的导航系统方式设置高层的规划方案。红色部分

虚线是各个动态障碍物实际运行中产生的轨迹。绿色虚线部分为选定主要研究对象的轨迹，并且研究对象还会在车身部分增添相应 `base_link` 坐标系，并通过绿色箭头部分（即其 `odometry`）来研究其各个时刻对应的速度方向。

实验假定场景中存在的障碍物都以阿克曼模型为约束，并选定一个移动移动机器人作为主要研究对象，观察其轨迹。如此设置实验的好处是，虽然只选定了一个机器人作为研究对象，但是其他机器人使用的是与选定代理一样的算法，可以最大程度模拟现实应用中出现的各种遭遇情况，并且选定的代理亦可作为其他实验对象的动态障碍物，多点研究，模拟情况种类丰富。并且由于实际使用中，路径的规划是本身自带静态避障的，故本仿真实验可以使用车模型一定程度替代实际场景上遭遇到的行人等物体。

实验设计主要分为两个类别、两种场景：1. 4 或者 8 动态障碍物的相向对穿实验、同向争点实验以及同向穿插实验。通过这三个实验分别验证现实场景中可能出现的几种穿越动态区域的情况。考虑到仿真环境地地图栅格分辨率为 $0.05m$ ，且环境中设置的机器人最大速度为 $0.3m/s$ ，故本文实验所取得时间窗口大小 $\tau = 3s$ ，即图中移动机器人在移动两个大栅格的情况下都不会发生与其他机器人的碰撞，这样取值是十分合理且具有代表性的。

4.4.1 实验一：4 动态障碍物遭遇实验

实验在同一无静态障碍物区域放置 4 个动态物体，此 4 物体即互相做障碍物也单独可称为研究主体。

1. 对穿实验

实验中选定的 ORCA 算法所需要的 V_{opt} 是 5.3 节局部规划器规划出的运动元路径所指向的速度方向。故此种选法是一种激进的 V_{opt} 选择策略。

上述图片展示了各动态障碍物面向终点并且对穿的过程，可以看到在中心的交汇处，由于 ORCA 算法对速度的调整，轨迹出现了一定程度的改变。下面展示更多情况下的实验室结果并配以选定的对象的速度调整图像，下图中不再展示运行过程，尽量多地展示实验结果：此实验结果中包含了车头朝向不指向目标点等的情况，从对穿实验可以看出，算法规划出的轨迹总是可以无碰撞且最优化地到达对面目标点，无论阿克曼的模型处于什么样的状态和朝向。并且在运动过程中会根据实际情况实时调整速度的方向。

2. 争点穿插实验

让 4 动态物体同向出发，向着对面的移动障碍物方向进发，过程中会出现动态障碍物从侧边出现的各种情况，实验结果展示如下：

上图中的 (b)、(c) 图可以明显从速度的方向指向变化中感受到 ORCA 算法模块对移动机器人的速度调整，以避免即将到来的与其他移动机器人发生碰撞。

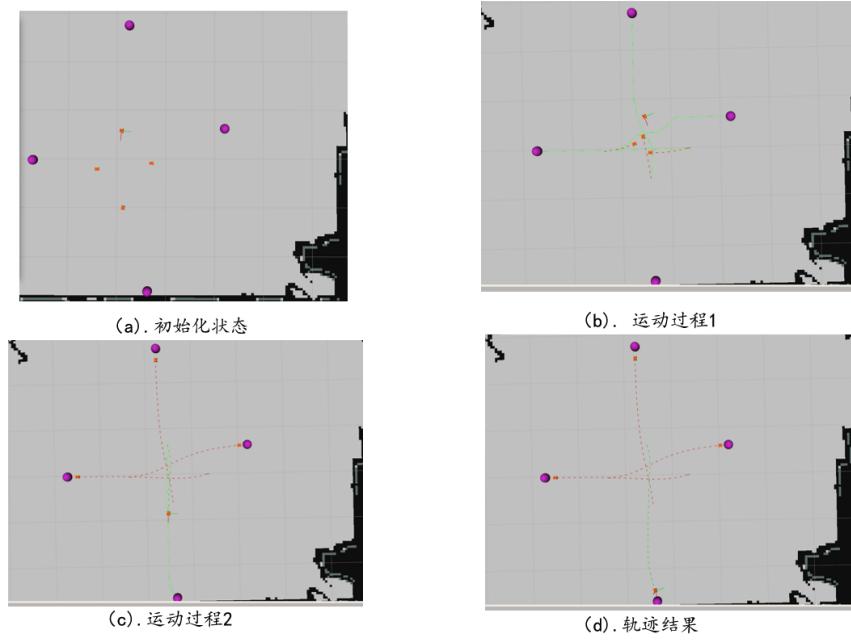


图 4.10 4 动态障碍物对穿实验 1

下图展示更多的实验结果以及将目标点数量减少，增加碰撞几率的实验结果。

4.4.2 实验二：8 动态障碍物遭遇实验

实验在同一无静态障碍物区域放置 8 个动态物体，此 8 物体即互相做障碍物也单独可称为研究主体，增多动态障碍物数量，可加大对算法的考研难度，具体做法类似实验一，此处展示实验的结果轨迹。

1. 对穿实验

由上图可以看出，增加了动态障碍物之后，移动机器人的路径规划显然难度大幅增加，图中目标点在右侧的两个移动机器人中的红色轨迹明显看出由于计算出实验选定的对象的速度与当前速度相冲，有明显的转向避让趋势。

下面的实验结果展示更多的为运动过程中选定的对象的速度变化与最终轨迹，不再展示相应的运动过程。通过 8 个动态障碍物的对穿实验可以看出，即使增加动态物后导致规划难度大大增加，但是本文提出的算法依然可以无碰撞且最优地规划出相应路径，并指引机器人到达目标点。

2. 争点穿插实验

争点穿插实验为增大对算法的考验力度，依然保持目标点的个数不变，此举会大大增加移动物体之间的碰撞几率，穿插实验的目标点位置设置以及运动过程如下所示：

本图片中展示的轨迹规划结果可明显看出，移动机器人虽有 A* 算法规划出全局轨迹，但在多个移动障碍物的速度障碍阻拦下，不得不调整当前行驶路径以

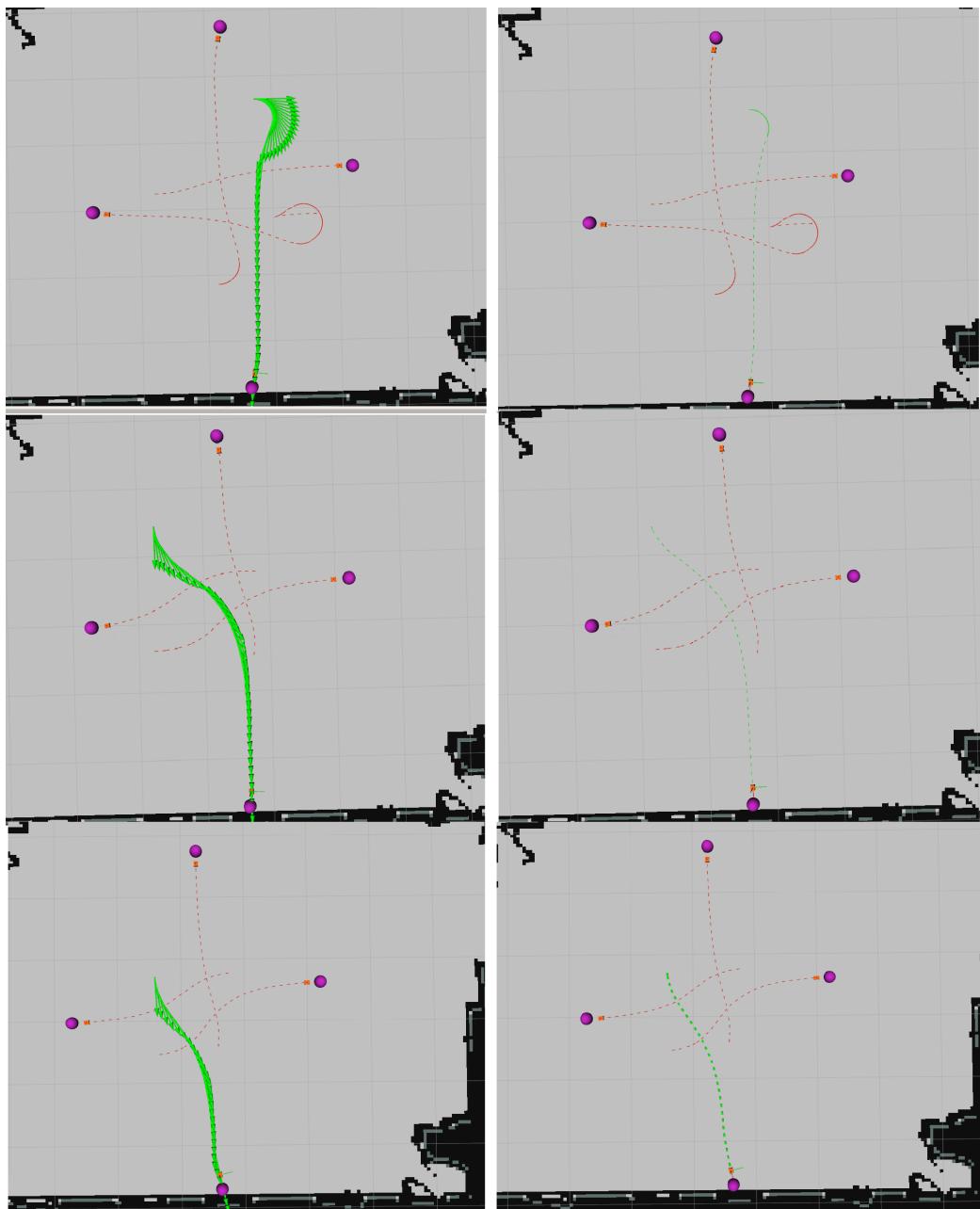


图 4.11 4 动态障碍物对穿实验 2 (左边表示速度方向变化, 右边表示最终轨迹)

避免碰撞，这便是本文提出的算法调控的结果。

以下实验结果将更多展示 8 动态障碍物之间的争点穿插结果，并且下面实验中将目标点数量调整至 2 个点，在哪增加动态障碍物数量的基础上，增加全局 A* 算法在地图上规划出的轨迹密集度，实验结果展示如下：

下面展示仅有两个目标点情况下的实验结果：

经过以上两轮多次实验，从实验结果中的速度变化以及轨迹图可以得出结论，本文提出的避障算法在和导航系统框架结合后可发挥出良好的动态避障效果。

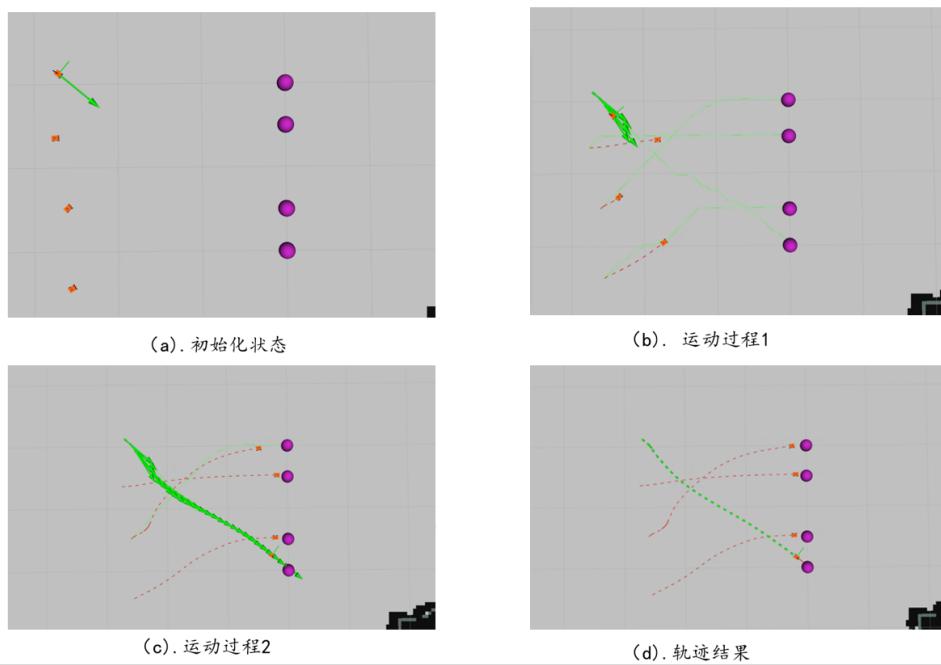


图 4.12 4 动态障碍物争点穿插实验 1

4.5 动态避障系统的实际场景实验

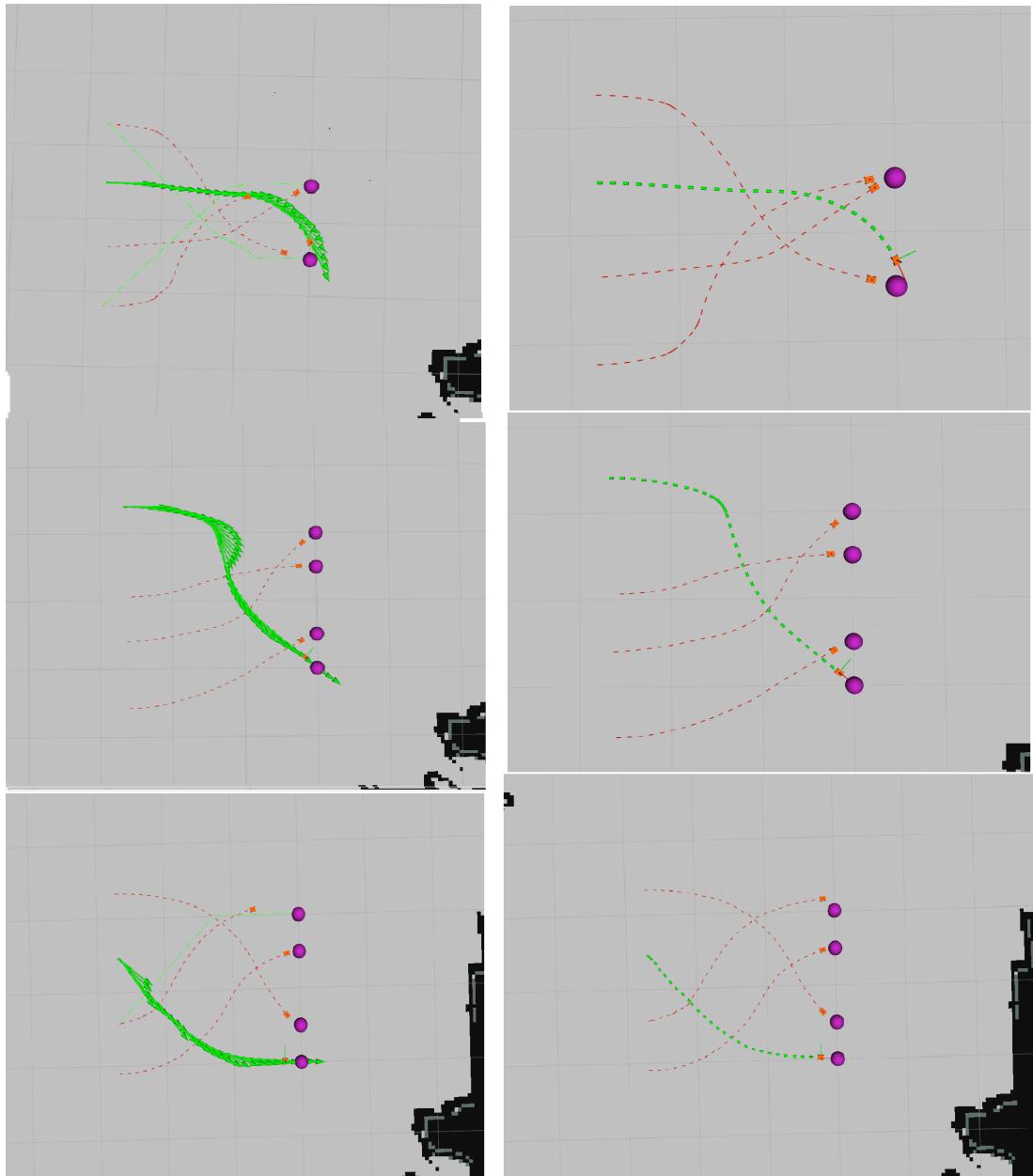


图 4.13 4 动态障碍物避让实验 2 (左边表示速度方向变化, 右边表示最终轨迹)

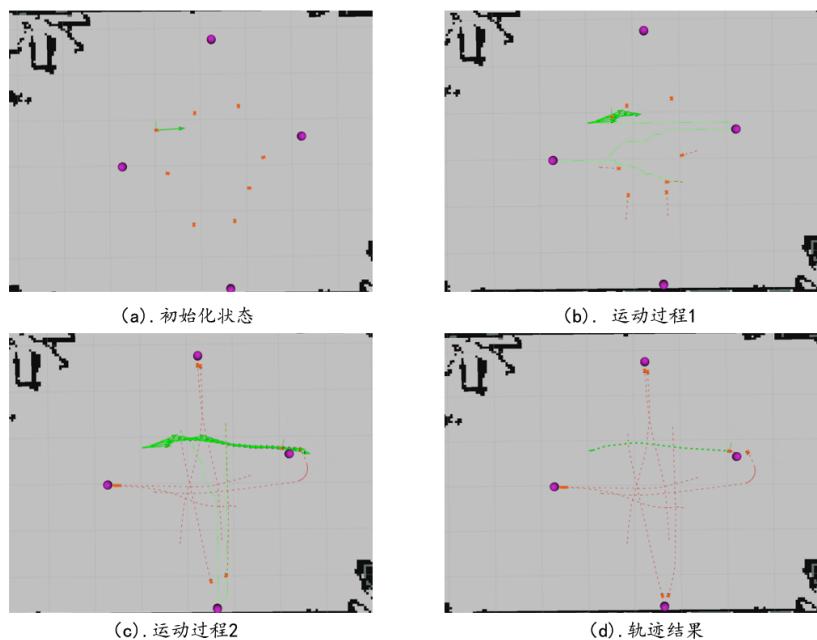


图 4.14 8 动态障碍物对穿实验 1

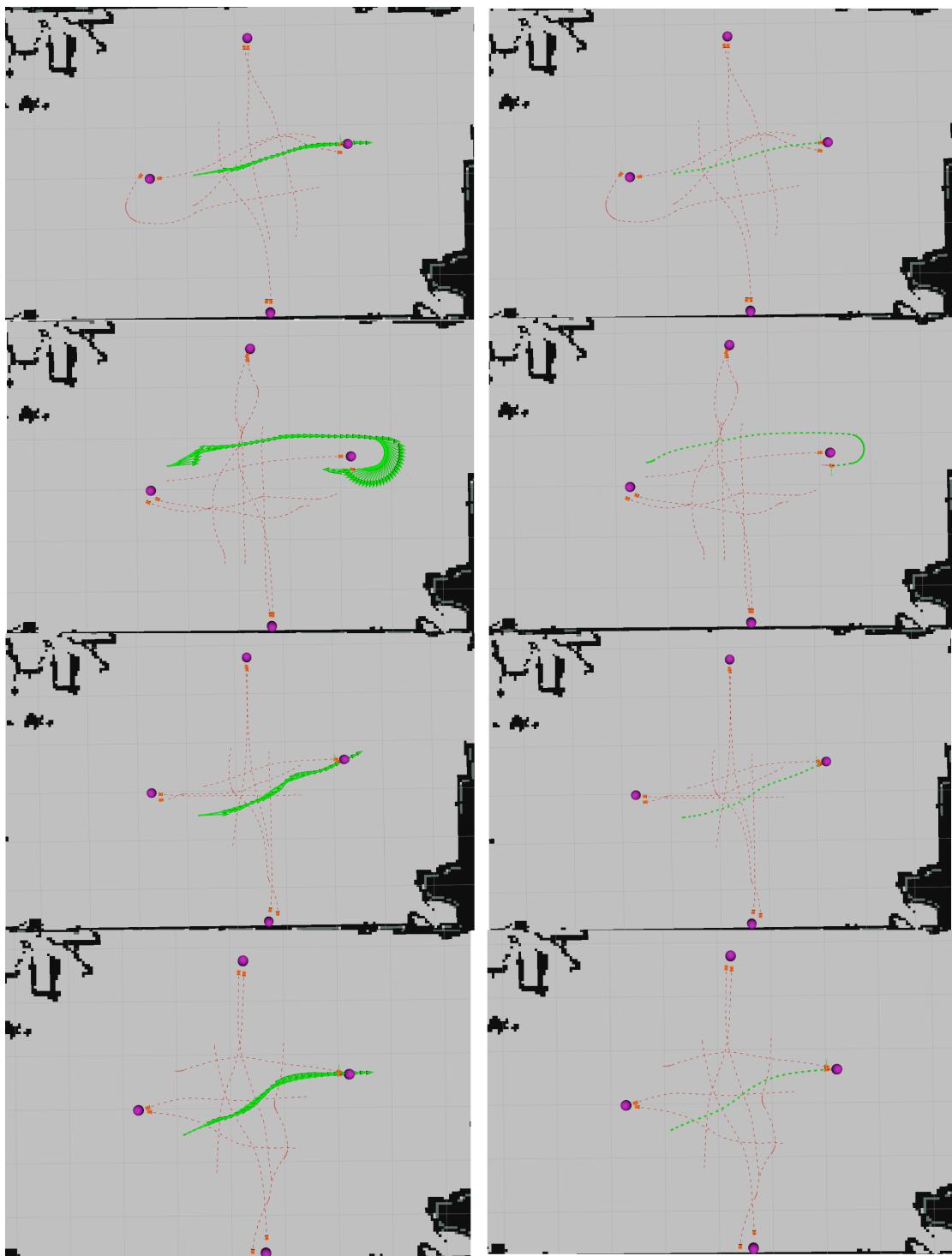


图 4.15 8 动态障碍物对穿实验 2 (左边图片为选定对象的速度变化, 右边为对应的轨迹图)

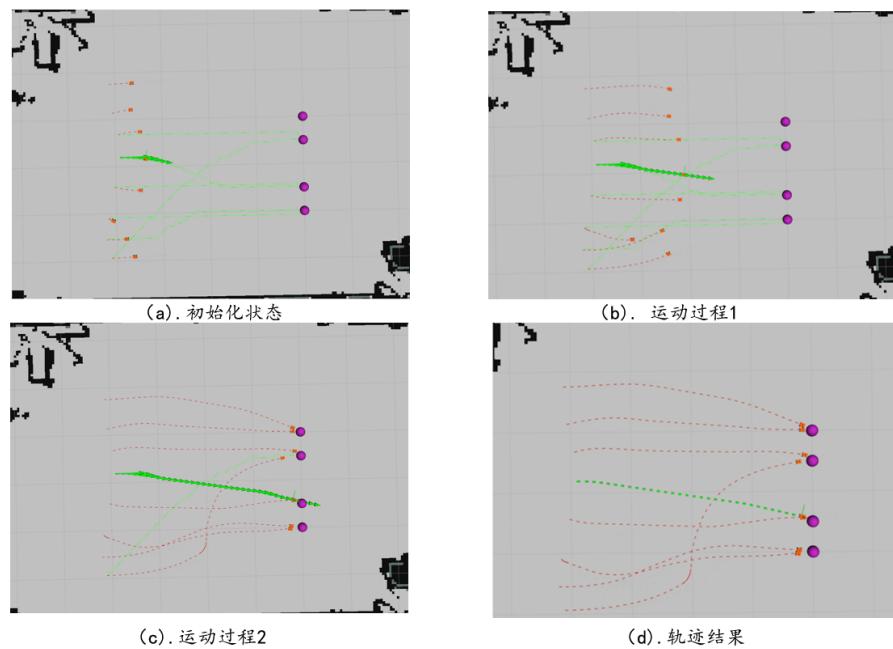


图 4.16 8 动态障碍物争点穿插实验 1

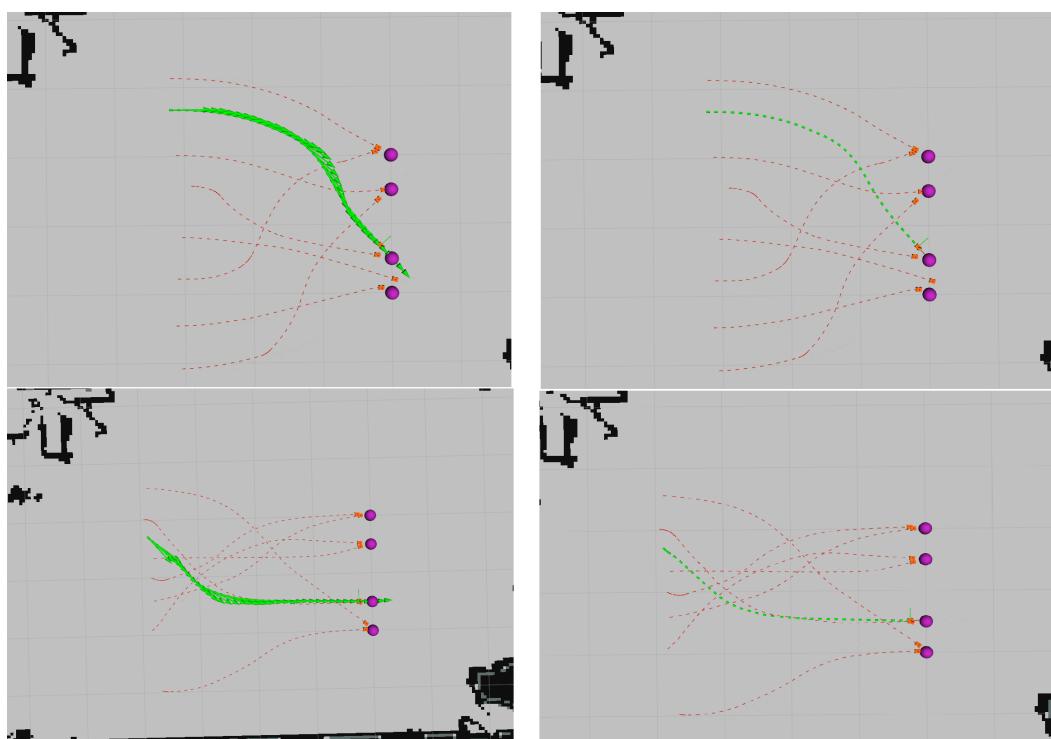


图 4.17 8 动态障碍物争点穿插实验 2 (4 目标点)

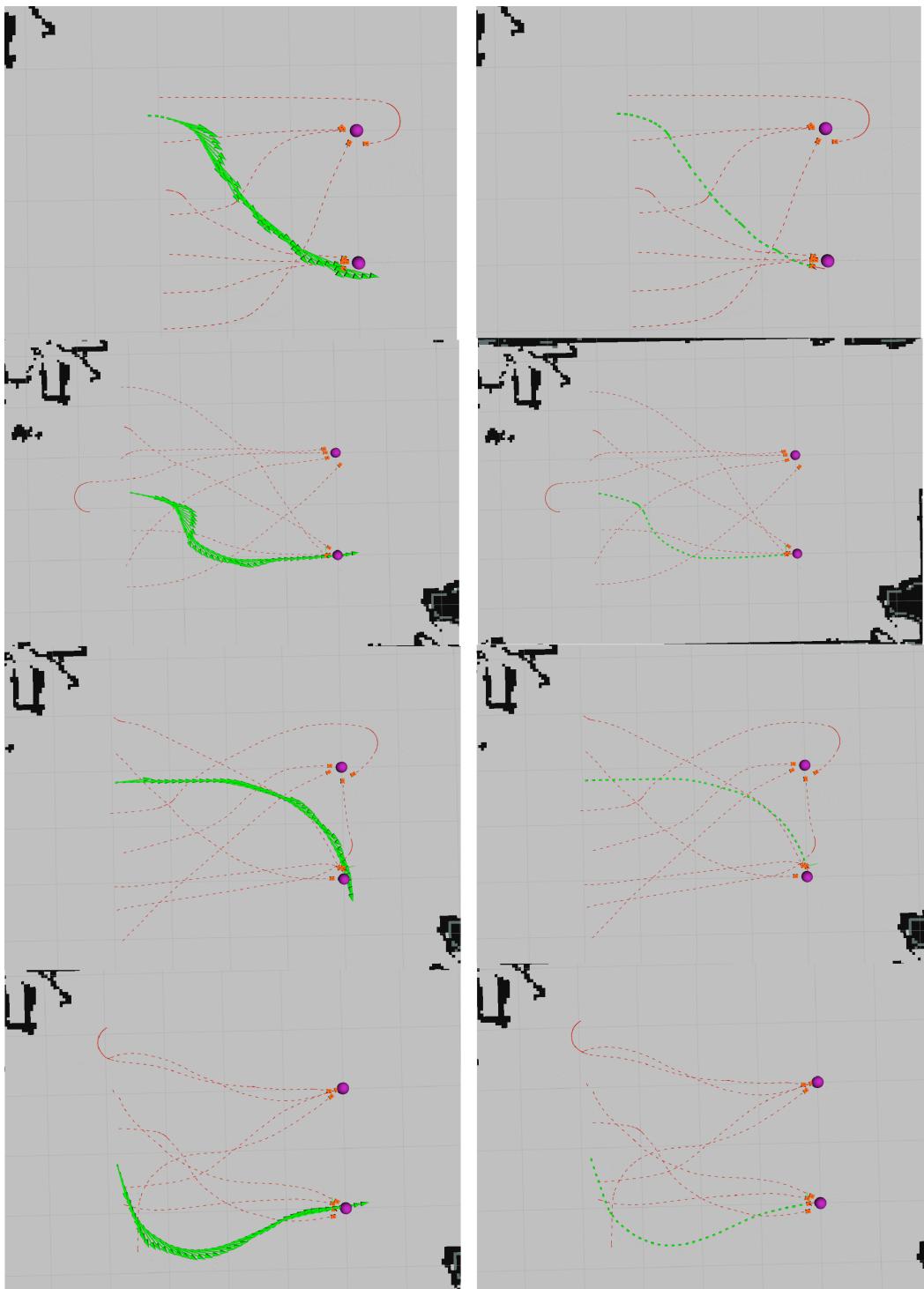


图 4.18 8 动态障碍物争点穿插实验 2 (2 目标点)

第5章 硬件系统、软件算法架构设计与系统信息流向

5.1 引言

上述3、4章节具体阐述了导航系统的两个层级之间的关系、区别以及具体设计方案。但是导航系统的工程实现还需要在方案设计的基础之上进行硬件平台的搭建与软件架构的设计。本章重点介绍根据上述两章的设计方案进行的硬件选型与相应的软件架构设计，并且给出系统中信息的流动方向，并且在实地测验中总结当前系统的实用优点与不足，为后续对系统的改进与园区内实用功能增添指明方向。

5.2 硬件系统设计

硬件系统主要分为两个方面来讲述，分别是硬件的选型与硬件的搭建方案。首先是硬件的选型首要依据是其与实验室底盘平台的适配性，其次是其成本。而硬件的搭建主要是根据当前所具备的硬件条件，如何有效规划这些硬件的使用与配合，保障系统在有限的硬件条件下，发挥出最佳的效果。小车的整体外形设计如下所示，展示小车的斜视与侧视图。

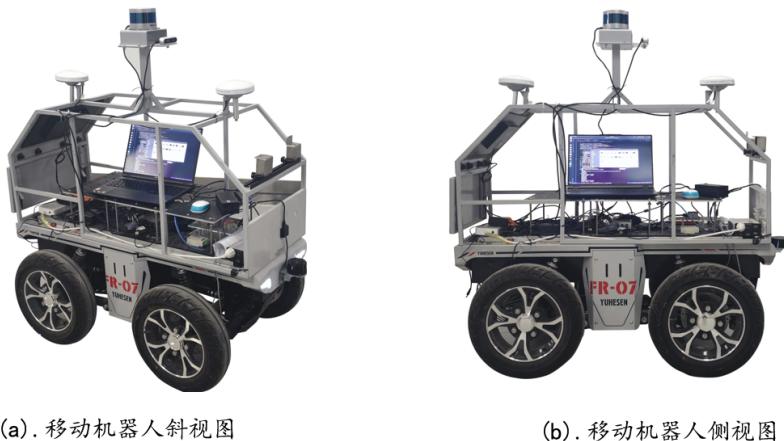


图5.1 移动机器人实物图

5.2.1 硬件选型

在移动机器人的硬件组成中，小零件相当多且繁杂，但是有几类硬件的重要性对于系统性能甚至于功能来说，至关重要。他们分别是执行计算的中央处理单元——计算平台；驱动执行部件——机器人底盘；环境感知传感器——激光雷达；高精度车载组合定位模块——RTK。首先最重要的计算单元我们根据系统算

法的要求，且根据目前实验室对算法的更新频率以及更新便捷性，选择了搭载了算力强大的 CPU 与 GPU 的 Legion 型号大算力计算机。校园环境下的使用证明，选型正确且留有了相当充足的计算裕量。

1. 智能无人平台计算核心 LENOVO LEGION 笔记本

计算核心是整个系统的重中之重，相当于人类的大脑，承担一切的计算工作，并且处理相应的输入信息，最终将处理得到的结果转化为控制指令，完成对车辆的控制。其性能参数为

此处插入表格

导航系统运行的计算机操作系统为 Ubuntu 18.04，具体通信平台为 ROS-Melodic-deskfull。以下是计算核心的实物图。



图 5.2 计算核心实物图

2. 煜禾森 FR07 阿克曼线控底盘

FR-07 是一款全能型机器人线控移动平台，它采用和汽车类似的结构—阿克曼结构，驱动后置，相对于差动结构的底盘，在普通路面上 FR-07 具有更快速的行走能力和较强的负载能力，同时对轮胎的磨损也更小，搭配整体桥式悬挂，能够通过减速带等常见障碍物，更适合长时间室外运行；并且，该底盘是基于车规

级 VCU 构建的自动驾驶底层，采用 CAN 总线管理，具有高精度、车规级、模块化等特点；通过搭载激光雷达、GPS、机械臂等上部模块和导航系统广泛应用于自动驾驶、无人巡检、物流、运输配送、科研以及各种新的需要移动底盘的应用探索中。线控底盘通过 CAN 盒与计算核心相连，接口类型为 USB，通信的数据类型为 yhs_can_msgs。具体控制方式是，计算核心输出 geometry_msgs::TwistStamped 类型消息，此消息会在相应的 fr07_control 被转化成 yhs_can_msgs 类型消息，从而达到计算核心和底盘的通信。我们的底盘自带 48V/20AH 的锂电池，可向车体持续供电 5h 以上，并且此电源可向外用设备，如激光雷达、RTK 等供电，供电标准为 5A-12V/15A-5V/4A。且在此基础上配备有可移动的 220V 稳压电源，用于户外向计算核心单独供电，保障整体用电安全性。

其通信相关的参数为：

此处插入表格

关于其他使用的性能参数为：

此处插入表格

下图是它的实物图。



图 5.3 底盘实物图（无雷达支架）

3. Velodyne VLP-16 激光雷达

不可否认目前已经出现了十分优秀的国产激光雷达产品以及厂商，例如禾赛科技、速腾聚创等。但是由于实验室平台搭建开始于 2-3 年前，当时条件下，Velodyne 的 VLP-16 型号激光雷达是十分适配实验室的移动智能机器人的，并且十分符合团队对园区场景下机器人“性能可观、成本可控”的设计理念。

此款雷达的性能参数为：

此处插入表格 性能参数：测量距离 100、重量 830g、360° 水平视场扫描、

$\pm 15^\circ$ 的垂直视场、竖直分辨率 2° 、水平分辨率在实验室小车上 10Hz 下为 0.2° 使用方法：智能无人平台计算平台为 LENOVO LEGION 笔记本，系统 Ubuntu18.04，ROS 版本：Melodic.

可以加入对雷达坐标系、点的存储等问题的分析描述

实物图如下：下图是它的实物图。



图 5.4 VLP-16 实物图

4. Ydlidar TG 30 激光雷达

底盘上安装 3D 激光雷达的位置决定了其对低矮且靠近移动机器人位置的物体扫描能力相当局限。而在系统移动过程中，充分考虑了园区场景内的特点，例如低矮的石凳、快速移动的小猫咪和流浪狗，在移动过程中针对此类低矮物体的避障，可以通过 2D 激光雷达补盲的方式。

2D 激光雷达的性能参数如下：

此处插入表格

2D 激光雷达的配置与实物图如下所示：

安装后，对 2D 激光雷达和 3D 激光雷达进行标定，并将不同坐标系下的点云整合到同一坐标系下，整合后的点云如图所示：图中亦展示了车后轮中心 `base_link` 与两个传感器坐标系之间的欧式变换关系。

当 3D 激光雷达与 2D 激光雷达组合后，系统的视野范围扩展到了车前方的低矮视野盲区，对低矮物体的扫描结果如下图所示：图中白色部分为 2D 激光雷达扫描点，彩色部分为 3D 激光雷达扫描点由图中的上两图可以看出，远处的低矮物体（蓝框），既有 3D 激光雷达的扫描点亦有 2D 激光雷达的扫描点，近处（黄框）的高处人身体有两者的扫描点，而低矮圆凳仅有 2D 激光雷达扫描线，同



(a). 2D 激光雷达安装位置斜视图

(b). 2D 激光雷达安装位置俯视图

图 5.5 2D 激光雷达安装实物图

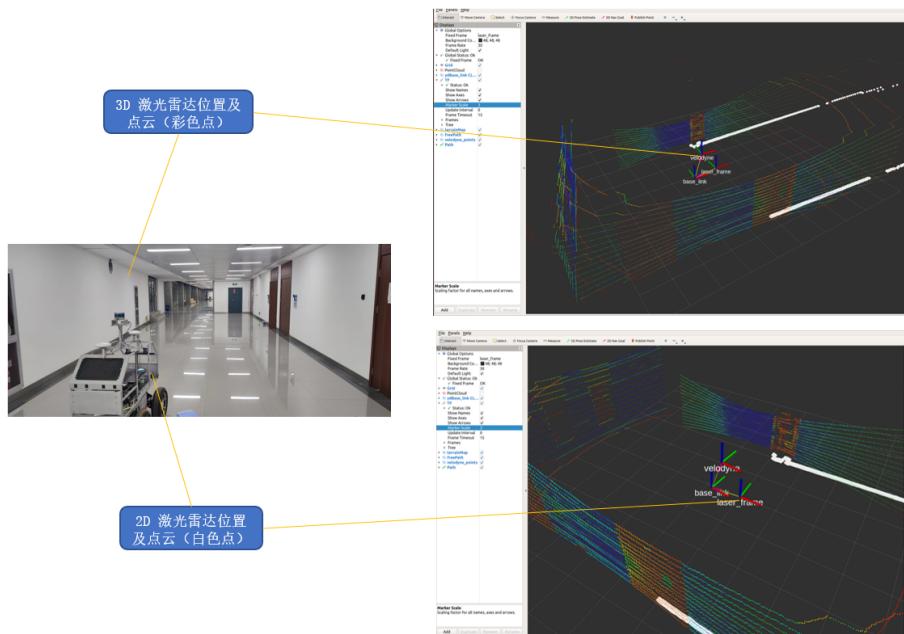


图 5.6 点云叠加图及其 tf 关系

时系统已经做出反应，对车体前方的路径做出了筛选。下三图更加佐证了这一功能，近处垃圾桶位置无彩色扫描点，仅有 2D 激光雷达的白色扫描线，并且相应的路径已有筛选。证明系统已经具备低矮盲区的扫描能力，不在对车身周围的低矮物体会有意外碰撞。

5. 车载组合高精度定位模块

定位是导航系统中尤为重要的一个模块，在具备了全局的先验地图之后，想要根据全局地图进行全局的路径规划，需要确定自身的实际位置。在依托于全局地图的导航中，我们通过车载组合高精度定位模块进行自身的实际定位。硬件的选型为中海达 Hi-Target RTK iNAV2 型号。其定位的性能参数为

此处插入表格

RTK 实物图如图所示：

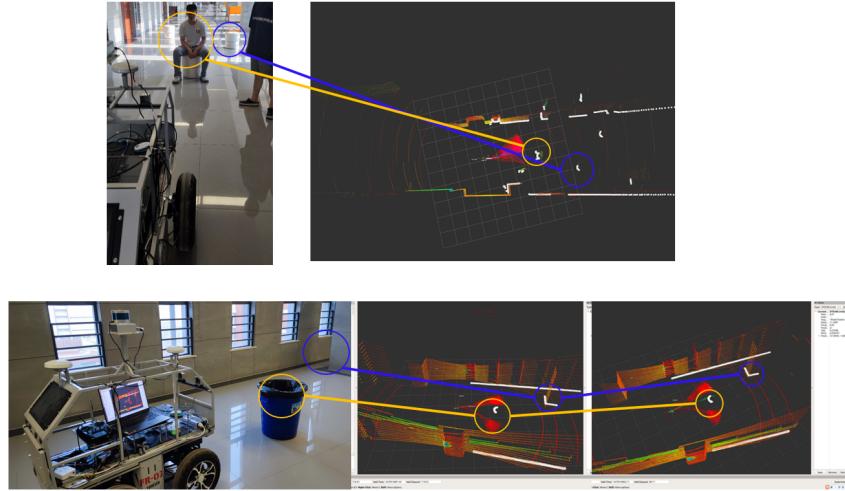


图 5.7 不同数量的低矮障碍物扫描视野情况图

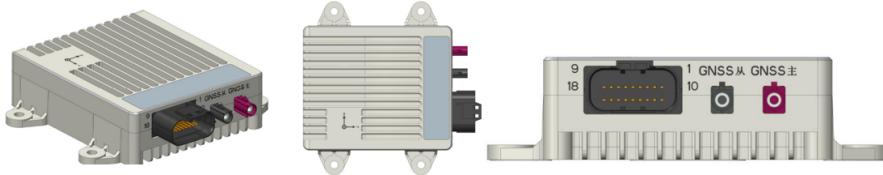


图 5.8 RTK 实物三视图

其中上述展示的为其主体，在使用过程中，RTK 包括了两个 GPS 信息接收终端，分别安装在移动机器人的前后位置，用以接收、校正 GPS 卫星定位数据。

5.2.2 硬件连接与信息交互

上述介绍了对硬件的选型，确定选型之后，便是考虑如何对硬件进行连接。

为了关系的清晰性与明确性，连接图的构建抛弃了所有的辅助链接设备，例如 CAN 盒等，只展示最核心的几大部件的相互连接关系。如下图所示：

计算核心在硬件中扮演的角色类似于人体中的大脑，无论是外界输入的信号，还是本身内部想要发出的执行指令都是通过计算核心在其中协调，并给出数据信号，完成整个过程。两个激光雷达类似于人类的眼睛，在整个导航系统中负责提取环境信息，并且将环境信息以点云的形式交给计算核心进行处理，类似于通过点云数据理解环境的过程，而计算核心又可以通过自身需求对采集点云的激光雷达发出一定要求，例如采集的频率；RTK 主要是通过卫星定位对移动机器人的当前位置进行更新，智慧生物可以自然地对自身所处位置进行评估更新修正，但是移动机器人无法做到这一点，所以 RTK 以计算核心下达的更新指令为依据，以需求频率对移动机器人的位置进行更新反馈，并在此过程中，将定位信息反馈给计算核心，再由计算核心调动参与移动机器人的其他功能；在得

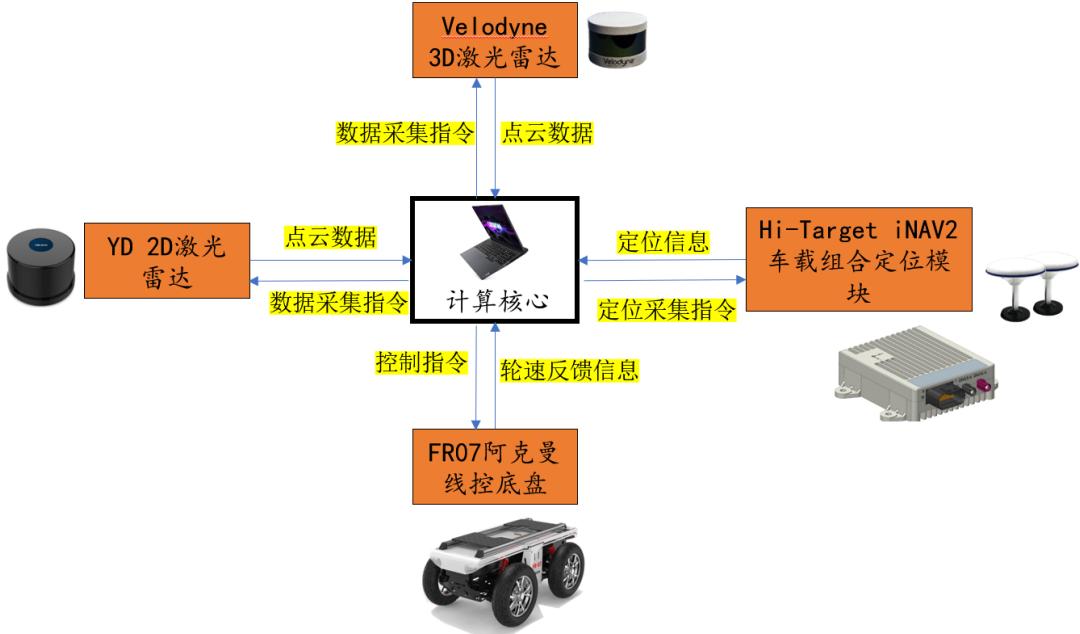


图 5.9 系统硬件连接与硬件数据流向图

到了环境信息以及自身在环境中的所处位置之后，经过一系列的导航算法处理，移动机器人会根据导航系统的安排，执行具体的动作到达相应位置，在这个过程中，主要是依靠底盘作为命令的执行器件，计算核心将控制命令发送给 CAN 总线，总线再将命令解读成阿克曼底盘可以理解的数字指令，交付执行。

如上流程便是各个主要器件参与导航系统的功能角色，接下来本文会在此基础上，结合第 3、4 章的导航系统设计部分，完成呈现整个导航系统的流程框图。

5.3 导航系统的实现及其信息流向

整个多层级导航系统的大致信息流向是高层导航规划出一系列路径点，底层导航根据环境信息以及自身定位，完成碰撞避免，安全执行一个个路径点的到达任务，当到达最后一个目标点时，整个导航任务完成。宏观上来看，多层级导航系统分为两个子系统，分别是高层系统和底层导航，两者之间通过局部路径点作为信息沟通的桥梁。宏观的框图如下：

在此整体的架构图基础之上，分别展开对底层导航和高层导航的架构图详细介绍。在宏观的系统架构之上，底层导航和高层导航之间除了少量关联之外，多数情况下功能是分别处理各自输入信息，因为对两个子系统做更加深入的了解。

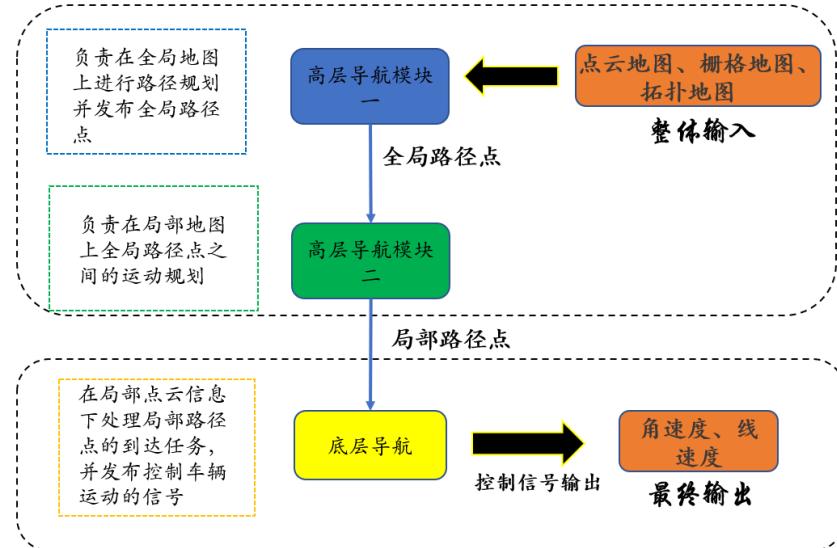


图 5.10 高层导航与底层导航信息流向图

5.4 底层导航的架构与信息流程

对于移动机器人的多层级导航系统而言，第一步是构建其底层导航系统，原因在于底层负责最基本的驱动执行与避障。无论何种形式的移动机器人，都需要从这个基础功能出发。

在第3章节中介绍了底层导航系统的设计思路与相应的算法原理，本节则是侧重于对思路的实现与具体运行过程中的相关信息解读。

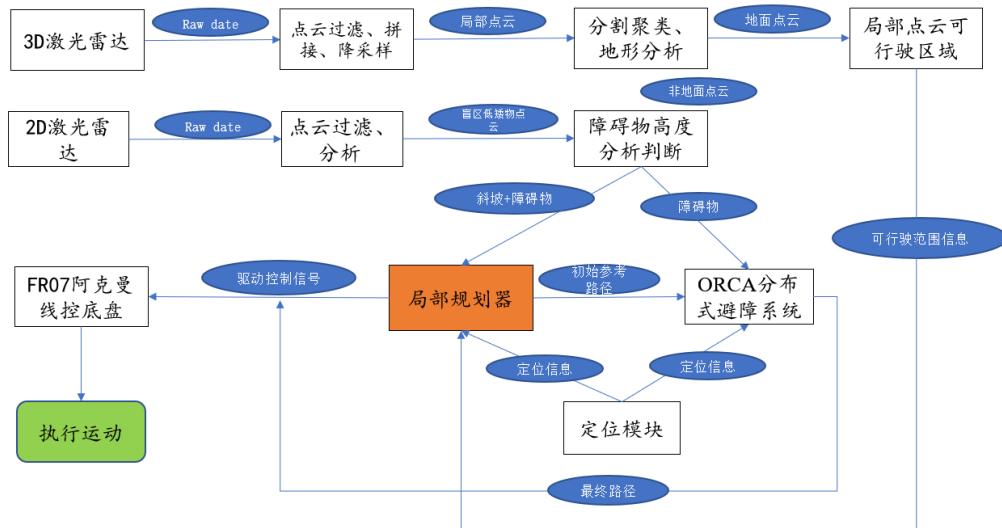


图 5.11 底层导航处理流程图

第6章 总结与展望

行文至此，已入终章。象士归隐，琴声渐廖。往生二十载光阴，从暮色村落到僻野小镇，喧嚣边城至繁华庐州；书声入耳垂髫堂坐到奋笔疾书只争太学，从河入海情系金陵至风华科大对酒蜀山；稚趣幼时到青涩年华，不惑有惑至而立将近；为求学尔。

路虽远，行则必达。

第7章 数学

7.1 数学符号和公式

《撰写手册》要求数学符号遵循 GB/T 3102.11—1993 《物理科学和技术中使用的数学符号》^①。该标准参照采纳 ISO 31-11:1992^②，但是与 TeX 默认的美国数学学会（AMS）的符号习惯有所区别。具体地来说主要有以下差异：

1. 大写希腊字母默认为斜体，如

$$\Gamma \Delta \Theta \Lambda \Xi \Pi \Sigma \Upsilon \Phi \Psi \Omega.$$

注意有限增量符号 Δ 固定使用正体，模板提供了 `\increment` 命令。

2. 小于等于号和大于等于号使用倾斜的字形 \leqslant 、 \geqslant 。
3. 积分号使用正体，比如 \int 、 \oint 。
4. 行间公式积分号的上下限位于积分号的上下两端，比如

$$\int_a^b f(x) dx.$$

行内公式为了版面的美观，统一居右侧，如 $\int_a^b f(x) dx$ 。

5. 偏微分符号 ∂ 使用正体。
6. 省略号 `\dots` 按照中文的习惯固定居中，比如

$$1, 2, \dots, n \quad 1 + 2 + \dots + n.$$

7. 实部 Re 和虚部 Im 的字体使用罗马体。

以上数学符号样式的差异可以在模板中统一设置。但是还有一些需要用户在写作时进行处理：

1. 数学常数和特殊函数名用正体，如

$$\pi = 3.14 \dots; \quad i^2 = -1; \quad e = \lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n.$$

2. 微分号使用正体，比如 dy/dx 。
3. 向量、矩阵和张量用粗斜体（`\mathsf`），如 \mathbf{x} 、 $\boldsymbol{\Sigma}$ 、 \boldsymbol{T} 。
4. 自然对数用 $\ln x$ 不用 $\log x$ 。

模板中使用 `unicode-math` 宏包配置数学字体。该宏包与传统的 `amsfonts`、`amssymb`、`bm`、`mathrsfs`、`upgreek` 等宏包不兼容。本模板作了处理，用户可以直接使用 `\bm`、`\mathsf`、`\mathscr`、`\upGamma` 等命令。关于数学符号更多的用法，参见 `unicode-math` 宏包的使用说明和符号列表 `unimath-symbols`。

^①原 GB 3102.11—1993，自 2017 年 3 月 23 日起，该标准转为推荐性标准。

^②目前已更新为 ISO 80000-2:2019。

7.2 量和单位

宏包 `siunitx` 提供了更好的数字和单位支持:

- 12 345.678 90
- 0.3×10^{45}
- kg m s^{-1}
- μm μm
- Ω Ω
- 10 和 20
- 10, 20 和 30
- 0.13 mm, 0.67 mm 和 0.80 mm
- $10 \sim 20$
- $10^\circ\text{C} \sim 20^\circ\text{C}$

7.3 定理和证明

示例文件中使用 `amsthm` 宏包配置了定理、引理和证明等环境。用户也可以使用 `ntheorem` 宏包。

定义 7.1 If the integral of function f is measurable and non-negative, we define its (extended) **Lebesgue integral** by

$$\int f = \sup_g \int g, \quad (7.1)$$

where the supremum is taken over all measurable functions g such that $0 \leq g \leq f$, and where g is bounded and supported on a set of finite measure.

假设 7.1 The communication graph is strongly connected.

例 7.1 Simple examples of functions on \mathbb{R}^d that are integrable (or non-integrable) are given by

$$f_a(x) = \begin{cases} |x|^{-a} & \text{if } |x| \leq 1, \\ 0 & \text{if } x > 1. \end{cases} \quad (7.2)$$

$$F_a(x) = \frac{1}{1 + |x|^a}, \quad \text{all } x \in \mathbb{R}^d. \quad (7.3)$$

Then f_a is integrable exactly when $a < d$, while F_a is integrable exactly when $a > d$.

引理 7.1 (Fatou) Suppose $\{f_n\}$ is a sequence of measurable functions with $f_n \geq 0$. If $\lim_{n \rightarrow \infty} f_n(x) = f(x)$ for a.e. x , then

$$\int f \leq \liminf_{n \rightarrow \infty} \int f_n. \quad (7.4)$$

注 We do not exclude the cases $\int f = \infty$, or $\liminf_{n \rightarrow \infty} f_n = \infty$.

推论 7.2 Suppose f is a non-negative measurable function, and $\{f_n\}$ a sequence of non-negative measurable functions with $f_n(x) \leq f(x)$ and $f_n(x) \rightarrow f(x)$ for almost every x . Then

$$\lim_{n \rightarrow \infty} \int f_n = \int f. \quad (7.5)$$

命题 7.3 Suppose f is integrable on \mathbb{R}^d . Then for every $\epsilon > 0$:

- i. There exists a set of finite measure B (a ball, for example) such that

$$\int_{B^c} |f| < \epsilon. \quad (7.6)$$

- ii. There is a $\delta > 0$ such that

$$\int_E |f| < \epsilon \quad \text{whenever } m(E) < \delta. \quad (7.7)$$

定理 7.4 Suppose $\{f_n\}$ is a sequence of measurable functions such that $f_n(x) \rightarrow f(x)$ a.e. x , as n tends to infinity. If $|f_n(x)| \leq g(x)$, where g is integrable, then

$$\int |f_n - f| \rightarrow 0 \quad \text{as } n \rightarrow \infty, \quad (7.8)$$

and consequently

$$\int f_n \rightarrow \int f \quad \text{as } n \rightarrow \infty. \quad (7.9)$$

证明 Trivial. ■

Axiom of choice Suppose E is a set and E_α is a collection of non-empty subsets of E . Then there is a function $\alpha \mapsto x_\alpha$ (a “choice function”) such that

$$x_\alpha \in E_\alpha, \quad \text{for all } \alpha. \quad (7.10)$$

Observation 1 Suppose a partially ordered set P has the property that every chain has an upper bound in P . Then the set P contains at least one maximal element.

A concise proof Obvious. ■

参 考 文 献

- [1] RAPHAEL B. Robot research at stanford research institute[R]. STANFORD RESEARCH INST CA, 1972.
- [2] NILSSON N J, et al. Shakey the robot[M]. Sri International Menlo Park, California, 1984.
- [3] ROUMELIOTIS S I, SUKHATME G S, BEKEY G A. Sensor fault detection and identification in a mobile robot[C]//Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No. 98CH36190): volume 3. IEEE, 1998: 1383-1388.
- [4] JONES J L. Robots at the tipping point: the road to irobot roomba[J]. IEEE Robotics & Automation Magazine, 2006, 13(1): 76-78.
- [5] QUIGLEY M, CONLEY K, GERKEY B, et al. Ros: an open-source robot operating system [C]//ICRA Workshop on Open Source Software. 2009.
- [6] WEIDINGER F, BOYSEN N, BRISKORN D. Storage assignment with rack-moving mobile robots in kiva warehouses[J]. Transportation Science, 2018, 52(6): 1479-1495.
- [7] HURST N, CLABAUGH C, BAYNES R, et al. Social and emotional skills training with embodied moxie[A]. 2020.
- [8] MORAVEC H P. The stanford cart and the cmu rover[J]. Proceedings of the IEEE, 1983, 71 (7): 872-884.
- [9] NUCHTER A, SURMANN H, LINGEMANN K, et al. 6d slam with an application in autonomous mine mapping[C]//IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004: volume 2. IEEE, 2004: 1998-2003.
- [10] CAO C, ZHU H, YANG F, et al. Autonomous exploration development environment and the planning algorithms[C]//2022 International Conference on Robotics and Automation (ICRA). IEEE, 2022: 8921-8928.
- [11] ZHU H, CAO C, XIA Y, et al. Dsvp: Dual-stage viewpoint planner for rapid exploration by dynamic expansion[C]//2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2021: 7623-7630.
- [12] CAO C, ZHU H, CHOSET H, et al. Tare: A hierarchical framework for efficiently exploring complex 3d environments.[C]//Robotics: Science and Systems: volume 5. 2021.
- [13] HUTTER M, GEHRING C, JUD D, et al. Anymal-a highly mobile and dynamic quadrupedal robot[C]//2016 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, 2016: 38-44.
- [14] ZIMMERMANN S, PORANNE R, COROS S. Go fetch!-dynamic grasps using boston dy-

- namics spot with external robotic arm[C]//2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021: 4488-4494.
- [15] ZHANG M X, WU J Y, WU X, et al. Hybrid evolutionary optimization for takeaway order selection and delivery path planning utilizing habit data[J]. Complex & Intelligent Systems, 2022, 8(6): 4425-4440.
- [16] VAN DEN BERG J, LIN M C, MANOCHA D. Reciprocal velocity obstacles for real-time multi-agent navigation[J]. Computer Animation and Virtual Worlds, 2008, 19(3-4): 207-221.
- [17] VAN DEN BERG J, GUY S J, LIN M, et al. Reciprocal n-body collision avoidance[C]//Robotics Research: The 14th International Symposium ISRR. Springer, 2011: 3-19.
- [18] GUY S J, CHHUGANI J, KIM C, et al. Clearpath: highly parallel collision avoidance for multi-agent simulation[C]//Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2009: 177-187.
- [19] HENNES D, CLAES D, MEEUSSEN W, et al. Multi-robot collision avoidance with localization uncertainty[C]//Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1. 2012: 147-154.
- [20] SNAPE J, VAN DEN BERG J, GUY S J, et al. Smooth and collision-free navigation for multiple robots under differential-drive constraints[C]//2010 IEEE/RSJ international conference on intelligent robots and systems. IEEE, 2010: 4584-4589.
- [21] ALONSO-MORA J, BREITENMOSER A, BEARDSLEY P, et al. Reciprocal collision avoidance for multiple car-like robots[C]//2012 IEEE International Conference on Robotics and Automation. IEEE, 2012: 360-366.
- [22] SNAPE J, GUY S J, VAN DEN BERG J, et al. Smooth coordination and navigation for multiple differential-drive robots[C]//Experimental Robotics: The 12th International Symposium on Experimental Robotics. Springer, 2014: 601-613.
- [23] ZHANG J, SINGH S. Loam: Lidar odometry and mapping in real-time.[C]//Robotics: Science and Systems: volume 2. Berkeley, CA, 2014: 1-9.
- [24] SHAN T, ENGLOT B. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain[C]//2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018: 4758-4765.
- [25] WAN G, YANG X, CAI R, et al. Robust and precise vehicle localization based on multi-sensor fusion in diverse city scenes[C]//2018 IEEE international conference on robotics and automation (ICRA). IEEE, 2018: 4670-4677.
- [26] DURRANT-WHYTE H, BAILEY T. Simultaneous localization and mapping: part i[J]. IEEE Robotics & Automation Magazine, 2006, 13(2): 99-110.
- [27] HART P E, NILSSON N J, RAPHAEL B. A formal basis for the heuristic determination of

- minimum cost paths[J]. IEEE transactions on Systems Science and Cybernetics, 1968, 4(2): 100-107.
- [28] LIKHACHEV M, FERGUSON D, GORDON G. Planning long dynamically feasible maneuvers for autonomous vehicles[C]//2008 IEEE International Conference on Robotics and Automation. IEEE, 2008: 528-533.
- [29] LAVALLE S M. Rapidly-exploring random trees: A new tool for path planning[C]//Robotics and automation, 2000. proceedings. ICRA'00. IEEE international conference on: volume 2. IEEE, 1998: 2666-2671.
- [30] KARAMAN S, FRAZZOLI E. Sampling-based algorithms for optimal motion planning[C]// 2011 IEEE International Conference on Robotics and Automation. IEEE, 2011: 961-966.
- [31] MUR-ARTAL R, MONTIEL J, TARDÓS J. Orb-slam2: an open-source slam system for monocular, stereo, and rgb-d cameras[J]. IEEE Transactions on Robotics, 2017, 33(5): 1255-1262.
- [32] QIN T, LI P, SHEN S. Vins-mono: A robust and versatile monocular visual-inertial state estimator[J]. IEEE Transactions on Robotics, 2018, 34(4): 1004-1020.
- [33] WISTH D, CAMURRI M, DAS S, et al. Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry[J]. IEEE Robotics and Automation Letters, 2021, 6(2): 1004-1011.

附录 A 补充材料

A.1 补充章节

补充内容。

致 谢

在研究学习期间，我有幸得到了三位老师的教导，他们是：我的导师，中国科大 XXX 研究员，中科院 X 昆明动物所马老师以及美国犹他大学的 XXX 老师。三位深厚的学术功底，严谨的工作态度和敏锐的科学洞察力使我受益良多。衷心感谢他们多年来给予我的悉心教导和热情帮助。

感谢 XXX 老师在实验方面的指导以及教授的帮助。科大的 XXX 同学和 XXX 同学参与了部分试验工作，在此深表谢意。

在读期间发表的学术论文与取得的研究成果

已发表论文

1. AAAAAAAA
2. AAAAAAAA
3. AAAAAAAA

待发表论文

1. AAAAAAAA
2. AAAAAAAA
3. AAAAAAAA

研究报告

1. AAAAAAAA
2. AAAAAAAA
3. AAAAAAAA