广东工业大学硕士学位论文

（工学硕士）

基于视觉 SLAM 的移动机器人室内未知环境探测研究

池鹏可

二〇一七年五月

分类号：　　学校代号：11845

UDC：　　密级：　　学　　号：2111415022

广东工业大学硕士学位论文

（工学硕士）

基于视觉 SLAM 的移动机器人室内未知环境探测研究

池鹏可

指导教师姓名、职称：　苏成悦　教授

学科(专业)或领域名称：电子科学与技术

学 生 所 属 学 院：　物理与光电工程学院

论 文 答 辩 日 期：　2017 年 5 月 28 日

A Dissertation Submitted to Guangdong University of Technology

for the Degree of Master

(Master of Engineering Science)

Research of Mobile Robot on Unknown Indoor Environment Exploration based on Visual SLAM

M.E.Candidate:Chi Pengke

Supervisor: Prof. Su Chengyue

May 2017

School of Physics & Opto-electronic Engineering

Guangdong University of Technology

Guangzhou, Guangdong, P. R. China, 510006

摘 要

随着移动机器人领域的发展，移动机器人能有效地在复杂危险的环境下执行任务，移动机器人的智能自主性得以提高。对未知环境进行探测建图并实时导航是移动机器人技术的研究难点。移动机器人在未知环境下根据自身位置估计和传感器数据，自主智能化实现自身定位和建立环境地图，这一过程称为同步定位与地图构建（Simultaneously Localization And Mapping，SLAM）。视觉传感器具有信息量丰富、轻量级、便宜等优点，将SLAM与视觉传感器相结合已成为机器人自主导航的研究热点。传统方法中更多采用扩展卡尔曼滤波或粒子滤波等滤波器方法来解决SLAM问题，但此类方法存在线性化及更新效率等问题，难以应用于创建大规模地图。随着SLAM相关高效求解方法的深入研究，基于图优化理论的SLAM技术能很好的创建大规模环境地图，并明显减少积累误差。

本文采用以华硕Xtion PRO为代表的RGB-D摄像头作为视觉传感器，同时获取室内环境的图像数据和深度数据，能够建立室内环境三维信息模型和用于移动机器人导航的二维栅格地图，主要研究工作如下：

1.本文仔细讲述了Xtion PRO相机获取RGB图像数据和深度图像数据的实现步骤，利用最小化残差平方得到相邻点云数据的相对位置关系，进而对点云数据进行配准与融合，实现了重建三维点云地图。

2.本文详细分析并比较了SIFT、SURF和ORB这三种特征点算法的提取特征点数目、运算时间以及正确率，最终使用了ORB算法实现特征提取与描述子计算。利用RANSAC算法与ICP算法相结合来求解相机运动估计的旋转矩阵 R 和平移向量 t 。

3.本文实现了基于图优化的视觉SLAM后端，构建了SLAM图优化模型，对基于图优化的 SLAM 问题进行了数学推导，使用g2o开源库来实现了图优化，最后利用BOVW闭环检测算法提高了移动机器人在未知场景环境下对突发状况的鲁棒性。

4.本文提出了一种基于边界的探测策略来实现移动机器人自主探测未知环境并构建地图，采用基于 $\mathbf{A}^*$ 算法的全局路径规划搜索最短行走路径和基于动态窗口法的局部路径规划进行实时避障。

利用慕尼黑工业大学(TUM)计算机视觉小组提供的标准测试RGB-D数据集和实际场景对本文的视觉RGB-D SLAM算法的各个阶段进行了测试与对比分析，还对移动机

器人基于视觉RGB-D SLAM算法进行了自主探测室内实际场景的实验，对实验数据进行整理与分析。实验结果证明了本文所提出的视觉RGB-D SLAM算法能够满足实时性需求，算法精度较高。

关键词：三维点云；ORB；图优化；闭环检测；RGB-D SLAM

Abstract

With the development of mobile robotics field, mobile robot can effectively perform tasks in a complex and dangerous environment, and intelligence and autonomy of mobile robot have been improved. The research difficulty of mobile robot technology is that detecting unknown environment for building map and real-time navigation. Mobile robot intelligently realizes own positioning and establishes environmental map based on their position estimation and sensor data in unknown environment, that is known as simultaneous localization and map building, SLAM. Because visual sensor has the advantages of rich information, lightweight and cheap, combining SLAM and visual sensor has become the research focus of autonomous navigation. The most traditional methods such as using extended Kalman filtering or particle filtering method to solve the problem of SLAM, but such methods create problems such as linearization and low update efficiency, and is difficult to build a large-scale map. With the in-depth study of the efficient solution, SLAM technology based on the theory of the graph optimization can preferably create large-scale environmental maps, and significantly reduce error accumulation.

This paper uses Asus Xtion PRO RGB-D camera as a visual sensor, to obtain the RGB image and depth image of the indoor environment, and establishing 3D model and 2D grid map for mobile robot navigation, the main research work is as follows:

1.This article details the implementation steps how the Xtion PRO camera gets the RGB image and depth image, and using least square method to calculate the relative position relations of adjacent point cloud data. And then matching and integrate the point cloud data, that can reconstruct the 3D point cloud map.

2.In this paper, a detailed analysis and comparison of the number of feature points, operation time and accuracy in the three feature point extraction algorithms including SIFT, SURF and ORB. And finally, we use ORB algorithm to extract feature and calculate the descriptor. Combining the RANSAC algorithm and the ICP algorithm can solve the rotation matrix R and the translation vector t of the camera motion estimation .

3.This paper realizes the visual SLAM back-end based on graph optimization. Firstly, graph optimization model of SLAM is constructed. And then the SLAM problem based on graph optimization is carried on the mathematical deduction, and using g2o open source library to realize the graph optimization. Finally, using BOVW closed-loop detection

algorithm improves the robustness when the mobile robot is in unknown environment scene.

4.This paper proposes a detection strategy based on frontier that can make mobile robot autonomously detect unknown environment and build the map. Using the global path planning based on $\mathbf{A}^*$ algorithm to search the shortest path and using the local path planning based on dynamic window approach to avoid obstacle in real time.

Using the standard test RGB-D data set of TUM computer vision group and the actual scene to test and analyze all stages of visual RGB- D SLAM algorithm in this paper. Also do a experiment that mobile robot automatically detect indoor actual scene based on visual RGB-D SLAM algorithm, and then sorting and analyzing the experimental data. The experimental results proved that the proposed visual RGB-D SLAM algorithm can meet the demand of real-time, and the algorithm has high precision.

Keywords: 3D point cloud; ORB; Graph optimization; closed-loop; RGB-D SLAM

目 录

CONTENT

第一章 绪 论

1.1 选题背景及意义

移动机器人是灵活性极高和自主能力极强的自动化综合体，可具有感知、规划、协同等与人相似的能力，涉及了传感技术、机械电子、自动化控制、计算机科学、人工智能等各个领域学科的研究。随着机器人领域的新一代技术的飞速发展，移动机器人能够在复杂危险的确定性环境下，代替或帮助人类完成一些难以进行的工作任务，如灾难救援、处理爆炸物、医疗护理等。此外，移动机器人也应用于服务业、娱乐业等便民生活方面。因此，移动机器人的各项技术已经得到世界各国研究人员的广泛关注。

始于1972年斯坦福大学研究所成功研发出第一台智能机器人Shakey[1]，掀起了机器人的历史新篇章。20世纪80年代美国安保机器人ROBART实现地面的自动搜索、清洁等基础工作。1997年美国NASA研制了第一台用于在火星上从事科学考察工作的自主式移动机器人，该移动机器人使用了视觉传感器和激光传感器。21世纪初美国iRobot机器人公司推出的扫地机器人Roomba[2]，具有自主避障和自主回充等功能，自此移动机器人开始逐渐步入人们的日常生活。近二十几年来，我国加强智能机器人技术的研究力度，成功研发出一系列工业机器人和特种机器人[3]。一些高校科研基地也取得了丰硕的研究成果，清华大学自主研发了THMR-V智能车，该移动机器人能在路面上自主行驶、自主避让和巡航[4]；上海交通大学的FRONTER-1自主移动机器人装有多种视觉传感器，操作性简单，反应速度快[5]；哈尔滨工业大学研制的服务机器人装备了机械臂和激光等多种传感器，可受远程控制，是一款真正物联网意义上的机器人[6]。中国机器人产业联盟统计数据显示，2015年国产机器人市场份额上升至32.5%，移动机器人也逐步走向产业化，涌现一批小米、科沃斯等品牌的全自动扫地机器人，还有用于餐饮、商场等场所的智能服务机器人。

移动机器人的自主导航技术是机器人领域的研究热点之一，Leonard和Durrant-Whyte首次提出要实现移动机器人在陌生环境下自主导航[7]，需要解决三个因素：“我在哪儿？”、“我周围环境是怎样的？”以及“我怎样到达目标位置？”。前两个因素表示移动机器人在环境中实时定位和创建地图，后一个因素表示移动机器人的避障和路径规划的功能。当移动机器人处于户外环境，一般GPS可以解决这两个因素，

但GPS无法解决室内移动机器人的定位和建图的问题。移动机器人通过自身一系列不同类型的传感器感知未知环境的信息，实现对环境进行构建地图并能够自我定位与导航，这一过程叫做即时定位与建图(simultaneous localization and mapping)，简称SLAM[8]。

SLAM的整体问题相当复杂，是一个鸡生蛋还是蛋生鸡的问题，主要是指机器人在未知复杂环境下根据环境地图进行确定自身位置，同时依据自定位增量扩展建立环境地图。SLAM第一次被Smith和Cheeseman提出之后[9]，经过许多学者几十年的研究，解决SLAM问题的方法得到巨大进展，各种SLAM相关的算法以及解决方案被提出，并且有一些已经投入商业应用。SLAM技术的研究主要经历基于滤波器方法和基于图优化方法这两个阶段。基于滤波器方法是假定已知机器人的观测数据与控制信息，估计机器人的当前姿态和用于观测路标位置的后验概率。基于滤波器方法有扩展卡尔曼滤波EKF、粒子滤波PF等方法[10,11]，尽管研究学者们结合其他理论对基于滤波器方法进行了诸多优化改进，但由于基于滤波器的SLAM方法的线性化问题，在长时间运动和大规模构建地图时存在更新效率低，更容易造成误差累积等问题。20世纪末Lu和Milios提出了基于图优化的SLAM的概念[12]，即是解决误差的最小二乘法问题，有些学者利用SLAM问题中信息矩阵的稀疏性[13,14]，大大降低基于图优化SLAM的计算复杂度，同时融合词袋模型、深度学习等新理论技术的闭环检测更好地解决了创建大规模环境地图过程中误差累积的问题。

1.2 三维重建的研究现状

三维重建技术涉及了几何图形学、模式识别、机器视觉等诸多学科，其目的是获取二维图像的深度、纹理等信息来恢复空间中物体的三维形状和位置信息。三维重建技术广泛应用在机器人导航、3D 打印、虚拟现实、医疗诊断等方面。立体视觉技术的发展和三维视觉传感器、激光雷达等硬件设备的成熟应用，使得三维点云模型的获取更加容易稳定，也使得日常生活物体数字化、三维场景重建成为可能[15]，也让物体模型显示自身具体细节信息。

目前，三维重建技术的研究主要针对两种方式：基于三维扫描技术和基于三维建模软件。基于三维建模软件的方式无法自动对三维物体场景进行三维重建，仅限于设计创作。Xtion 和 Kinect 这类的 RGB-D 相机的出现，使得三维扫描技术成为低成本、轻便的三维重建技术，国内外研究学者根据 RGB-D 相机的深度信息进行了实际场景三维重建的研究。国内杨鸿等人[16]提出一种低成本 Kinect 相机的三维地图创建方法，有效

解决了移动机器人在未知室内环境下的三维感知问题。为了将三维重建技术应用在移动机器人上，贾松敏等人[17]针对室内复杂环境三维建模问题，提出一种移动机器人快速三维同时定位与地图创建 SLAM 方法。针对 RGB-D 相机进行三维重建的精度问题，梅峰等人[18]以 RGBD-SLAM 算法为基础，将深度图像信息加入帧间配准算法，提高了帧间配准算法的准确性与鲁棒性，也提出了一种指数权重函数有效地降低深度图像畸变对三维重建的影响。国外有研究学者[19-21]对在畸变矫正、特征点提取与匹配以及全局优化这几个环节涉及的算法进行改进，提高了三维点云地图的精确性。微软研究院的研究人员研发出一款 KinectFusion 系统，此系统利用 RGB-D 相机 Kinect 融合不同时序的深度信息，逐步完善重建室内目标物的三维模型，对重建虚拟场景实现了一定的人机交互[22]。Kainz 等人[23]基于 Kinect Fusion 系统改进相关算法，并提出一种能减少噪声干扰的物理设备和校准方法，同时使用多个 RGB-D 视觉传感器重建出高质量的三维模型，极大减少系统误差。Alexiadis 等人[24]研发了使用多个 RGB-D 相机 Kinect 的实时三维重建系统，提出了一种能够用于精准的、完整的移动前景目标三维重建的方法。

移动机器人在危险未知环境下进行探测三维重建，更能清楚了解到自身所处的环境信息，实现更好的导航效果和应急处理。本文主要介绍基于 RGB-D 视觉传感器的三维重建的研究，并应用在移动机器人自主导航上。

1.3 视觉 SLAM 概述及研究现状

由于视觉传感器具有成本低、质量轻、图像信息量大等优点，视觉 SLAM 是目前机器人导航方面的研究热点之一。移动机器人通过视觉传感器观测周围环境目标位置，从未知起点开始采用增量式创建环境地图，同时利用生成的环境地图进行实时定位，而且环境地图能有效地表示障碍物轮廓模型。因使用视觉传感器数目不同，视觉 SLAM 分为基于单目视觉方法、基于双目视觉方法和基于多目视觉方法[25]。一个完整稳定有效的视觉 SLAM 过程包括视觉传感器获取环境信息，环境图像特征提取与配准，视觉传感器的运动估计，后端数据优化以及生成地图和运动轨迹这些环节。根据建模方案可以分为基于滤波器和基于图优化两大类。

1.3.1 基于滤波器的视觉 SLAM 研究现状

基于滤波的 SLAM 是 SLAM 研究历史上最早解决 SLAM 问题的方法，20 世纪 90 年代初 Smith 等人最早提出基于卡尔曼滤波求解 SLAM 问题[26]。21 世纪初 Chiuso 等人[27]利用单目视觉信息，运用扩展卡尔曼滤波 EKF[28]以及增量式地图的方法解决 SLAM 问题。卡梅隆大学研究者使用粒子滤波器实现了一种 Fast SLAM[29,30]，避免了卡尔曼滤波中的计算复杂度和数据关联错误敏感度。Grisettiyz 等人[31]采用一种自适应的粒子预测分布方法，减少粒子数目，进而降低维护地图时的存储空间。Sibley 等人[32]提出一种滑动窗滤波器 SLAM，即是在滑动窗口中不仅保留当前位姿，还包括之前一段连续的位姿，以提高滤波器算法的精确度。国内有学者使用平方根容积卡尔曼滤波计算 SLAM 后验概率密度，以减小线性化误差，提高了机器人 SLAM 定位精度[33]。宋宇等人[34]提出平方根容积 Rao-Blackwillised 粒子滤波 SLAM 算法，有效解决大尺度环境中移动机器人 SLAM 问题，减小 SLAM 非线性模型线性化误差，提高了 SLAM 精度和计算效率。

1.3.2 基于图优化的视觉 SLAM 研究现状

基于滤波器的 SLAM 的线性化点不一致，无法重线性化，导致信息矩阵秩增加，对后续累积误差的影响十分严重。对于观测数量为 N 的环境地图，滤波器方法计算复杂度为 $O(N3)$，图优化的方法计算复杂度为 $O(N)$，图优化方法相对于滤波器方法具有更高的计算效率[35]。最早 1997 年 Lu 和 Milios 提出通过图优化来解决 SLAM 问题[36]，但当时研究者们认为此方案的计算非常复杂因而没被广泛应用。后来有研究者发现 SLAM 问题中的视觉地标只存在一小段的视觉观测图像中，可知求解优化问题时的偏导数矩阵是稀疏矩阵，因而大大降低了优化求解问题的计算量，使得高效求解基于图优化的 SLAM 问题成为可能[37]。Thrun 等人基于前人的研究基础上提出一种 Graph SLAM 方法[13]，利用稀疏化降低计算量，通过位姿间的约束绘制地图，进而简化为一个最小二乘估计问题。牛津大学 Klein 等人[38]首次提出一种基于关键帧光束平差法 BA 的单目视觉 SLAM 系统，名为 PTAM。Mur-Artal 等人[39]基于 PTAM 的算法框架提出一种 ORB-SLAM 系统，利用 ORB 描述子进行特征匹配和重定位，通过方位图优化闭环回路。

1.4 本文主要内容

本文利用 RGB-D 视觉传感器实现了一种视觉 SLAM 算法，有效地实现了移动机器人的即时定位与地图构建，提出了一种基于边界的探测策略来实现移动机器人自主探测室内未知环境，构建室内环境 3D 点云地图和用于移动机器人导航的 2D 栅格地图。本文各章节安排如下：

第一章为绪论，简要介绍了本课题研究的背景与意义，然后讲述了三维重建的国内外研究现状，对视觉 SLAM 及其国内外研究现状进行综述，阐述了本文研究工作的主要内容和论文结构。

第二章为三维点云重建的原理，先是详细介绍了 RGB-D 传感器的硬件结构，描述了华硕 Xtion PRO 相机的标定方法及步骤，接着介绍了 Xtion PRO 相机获取 RGB 图像数据和深度图像数据的过程，实现对点云数据进行去噪处理以及配准与融合。

第三章为视觉 SLAM 前端的实现，首先简要介绍视觉 SLAM 前端的实现流程，详细介绍了在特征点提取方面的 SIFT 算法、SURF 算法和 ORB 算法的原理，采用 ORB 算法来提高视觉 SLAM 的实时性。着重讨论了图像间特征点对应关系来求解相机运动估计。

第四章为视觉 SLAM 后端的实现，详细介绍了基于卡尔曼滤波和基于粒子滤波的 SLAM 的工作流程，着重叙述了构建 SLAM 图优化模型，使用 g2o 开源库来实现图优化，最后叙述了基于 BOVW 模型的闭环检测方法的实现流程。

第五章为移动机器人自主探测的实现，本章首先阐述了实现移动机器人自主探测的需求和大致过程，分别详细介绍了全局路径规划和实时避障的局部路径规划的实现步骤，最后详细描述了基于边界的自主探测策略。

第六章为实验与分析，先介绍了实验环境场景以及实验平台配置，详细描述了利用标准测试数据集和实际场景对本文的视觉 RGB-D SLAM 算法进行了测试与评估，以及叙述了移动机器人基于视觉 RGB-D SLAM 算法进行自主探测室内实际场景实验。

第二章 三维点云重建的原理

2.1 RGB-D 传感器介绍及标定

目前市场上主流的 RGB-D 传感器相机有微软的 Kinect 和华硕的 Xtion PRO，两者核心技术都是来自以色列 PrimeSense 公司的 PS1800 芯片。在媒体娱乐方面，RGB-D 传感器使用 3D 人体运动捕获算法实现动作捕捉和手势识别等功能，可以让人们无须手持控制器，直接通过手势操纵游戏。在三维重建方面，RGB-D 传感器可以同时获取室内环境的彩色图像信息和深度图像信息，相比立体相机和 TOF 相机[40]，具有轻便、便宜、信息完整等优点[41]。

2.1.1 RGB-D 传感器硬件结构

主流 RGB-D 传感器实物图如图 2-1 和图 2-2 所示，传感器主要由一颗 PS1800 芯片、一个 RGB 摄像头、一个 IR 摄像头和一个 IR 投射器组成。PS1800 芯片是一个多感应系统级的芯片，具有极高的并行运算能力，主动投射红外光源从而对图像进行编码，能够同步获取深度图像和彩色图像。RGB 摄像头用于捕捉彩色图像信息，IR 摄像头和 IR 投射器组成了三维结构光深度传感器，用于捕捉深度图像信息。微软 Kinect 和华硕 Xtion PRO 都是结构光三维扫描设备，从规格、感应距离等参数对比 Kinect 和 Xtion PRO，对比结果如表 2.1 所示。经对比可知，华硕 Xtion PRO 具有尺寸体积小和即插即用 USB 设计的优点，因此本文使用的 RGB-D 传感器是华硕 Xtion PRO。

图 2-1 微软 Kinect

Fig.2-1 Microsoft Kinect

图 2-2 华硕 Xtion PRO

Fig.2-2 ASUS Xtion PRO

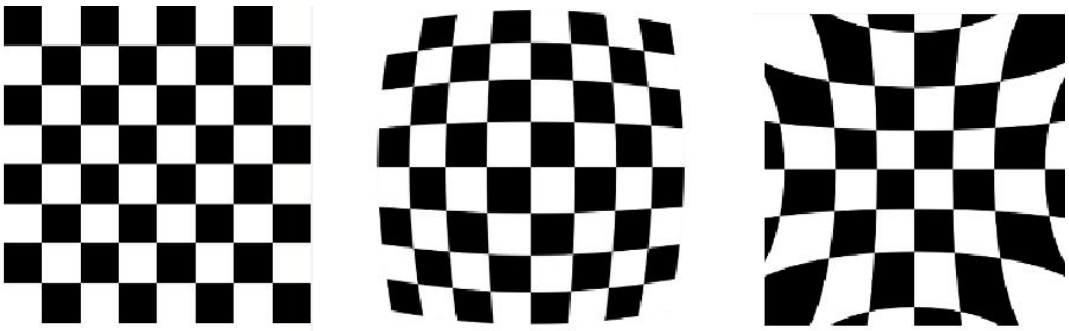
表 2-1 Xtion Pro 与 Kinect 指标参数对比

Table 2-1 Comparison of parameters of Xtion Pro and Kinect

指标	Xtion Pro	Kinect
长	18cm	28cm
宽	3.6cm	6cm
高	5cm	7.5cm
镜头部分高	2.6cm	4cm
感应距离	0.8<x<3.5	1.2<x<3.5
所包含传感器	RGB 摄像头、IR 摄像头和 IR 投射器	RGB 摄像头、IR 摄像头和 IR 投射器
有效视角	70°	水平：57° 垂直：43°
接口	USB2.0	外界电源+USB2.0

2.1.2 华硕 Xtion PRO 相机标定

由于透镜形状会产生径向畸变，成像过程中产生的径向畸变使物体投射在成像平面上存在误差，如下图 2-3 所示。相机标定可以对畸变的图像进行校正，减少图像失真。相机标定的目的是为了求得相机的内参数矩阵（即是光心和焦距），通过内参数矩阵使图像像素点坐标和三维点坐标之间相互转换。目前相机标定方法主要有传统相机标定法、主动视觉相机标定方法、相机自标定法[42]。而本文华硕 Xtion PRO 相机标定使用的是现在主流的张正友标定法，此方法介于传统相机标定法和相机自标定法之间，既降低对设备的高要求，又不影响应有的精度。



(a) 正常图像 (b) 桶形畸变图像 (c) 枕形畸变图像

(a) Normal image (b) Barrel distortion image (c) Pincushion distortion image

图 2-3 畸变图像

Fig.2-3 distortion image

华硕 Xtion PRO 相机的摄像头采用了针孔模型，如下图 2-4 所示，其中 $F_c - X_cY_c$ 为 Xtion PRO 相机坐标系，O-xy 为图像坐标系，空间 3D 点坐标与其投影到图像中的 2D 点坐标的相互转换关系：

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.1}$$

其中，s 为尺度因子，(u,v)为图像坐标系下的点坐标，(X,Y,Z)为世界坐标下 3D 点坐标，$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$为相机的内参矩阵，$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$为相机外参矩阵，$f_x$ 和 f_y 为焦距，c_x 和 c_y 为光心，即是基准点，通常位于图像中心。

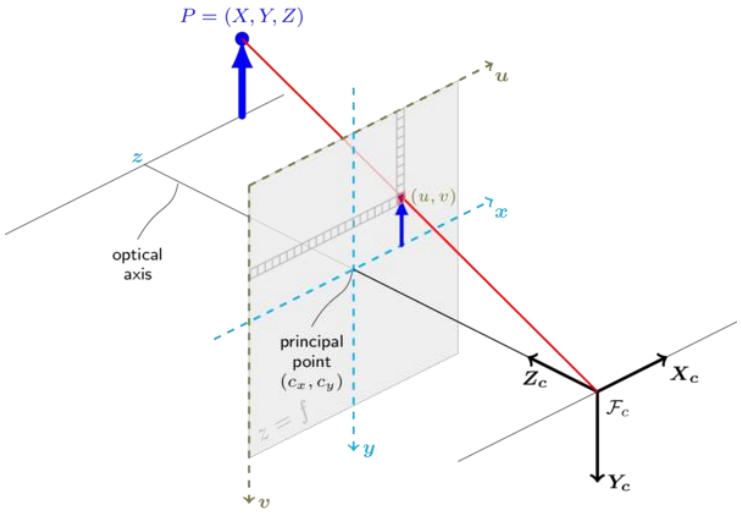


图 2-4 针孔模型

Fig.2-4 The pinhole camera model

相机内参矩阵反映的是相机坐标系与图像坐标系之间的变换关系，相机外参矩阵反映的是世界坐标系与相机坐标系间的变换。相机的运动估计是通过空间三维特征点的配准得到，特征点坐标是基于相机坐标系计算得到的，无法直接计算特征点在世界坐标系中的位置，所以需要对 Xtion PRO 相机进行标定得到内参矩阵。Xtion PRO 相机标定的具体流程如下：

(1)使用一张 6*7 规格的棋盘图，黑白相隔，每个正方形格子长 108mm。如下图 2-5 所示。

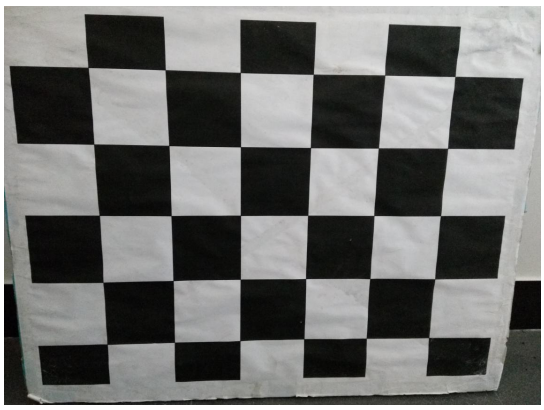


图 2-5 棋盘图

Fig.2-5 The board figure

(2)启动 ROS 下用于相机标定的脚本程序 cameracalibrator.py ，然后不断地移动棋盘，使相机从不同角度和不同距离获取棋盘的图像信息，以致得到 20 帧图像，如下图 2-6 所示。

图 2-6 不同角度下的棋盘图像

Fig.2-6 The chessboard images under different angles

(3)检测到图像的特征点如下图 2-7 所示。

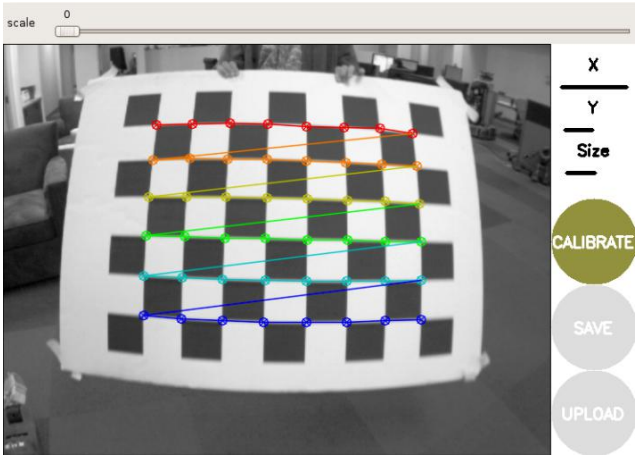


图 2-7 检测的特征点

Fig.2-7 The detected feature points

(4)经标定得到了 Xtion PRO 相机的内参矩阵 K，所下式(2.2)所示。

$$K=\begin{bmatrix}430.22 & 0.00 & 306.70\\ 0.00 & 430.53 & 227.22\\ 0.00 & 0.00 & 1.00\end{bmatrix} \tag{2.2}$$

2.2 图像数据获取与处理

非盈利组织发起了一个 OpenNI（Open Natural Interaction，开放式自然交互），其目的是提高应用软件与交互设备间的协同能力以及用户界面的交互性[43]。OpenNI 是一个多语言的、跨平台的开源体感框架，其定义的标准 API 可以编写通用自然交互应用，方便不同设备接口间更容易进行通信协作。OpenNI 的标准 API 能方便软件工程师直接利用最原始的数据格式进行算法开发，无需关心数据是从哪些设备产生，也使得硬件设备制造商无需考虑是否兼容上层应用程序。因此，我们可以利用 Xtion PRO 相机的 API 和 OpenNI 框架库实时获取室内场景的深度图像信息和彩色图像信息，然后使用中间件处理图像数据，最后输出能被上层应用程序直接使用的数据[44]。其中，如下图 2-8 所示，OpenNI 框架层次分为三层：

第一层是应用程序，开发人员可基于 OpenNI 框架编写应用软件。

第二层是设备接口层，即是 OpenNI 框架本身，将硬件部分和应用软件程序部分相连接。

第三层是硬件设备层，能获取视频和音频数据的支持 OpenNI 框架的各种硬件设备。

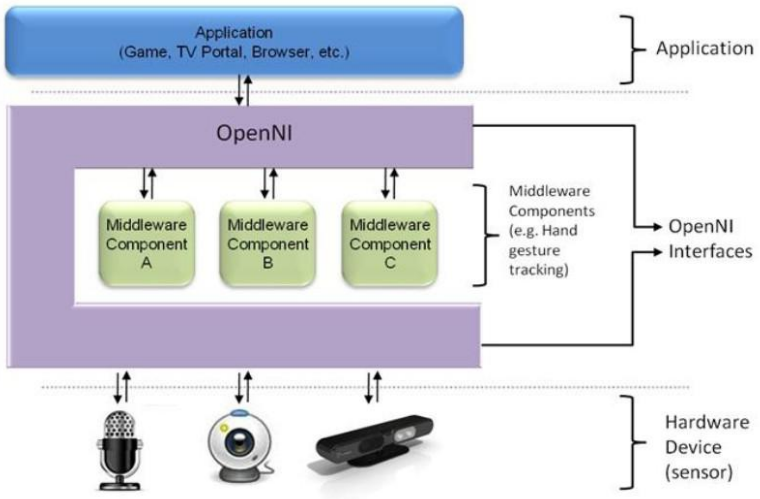


图 2-8 OpenNI 框架图

Fig.2-8 The frame of OpenNI

2.2.1 获取图像数据

Xtion PRO 相机的 IR 投射器与 IR 摄像头相结合可以生成三维数据，IR 摄像头获取到的数据，包括了深度图像坐标系的 X、Y、Z 的坐标值，Z 轴上的是深度距离。由于受到相机方位、视角范围以及物体形状等限制，我们需要从多个角度对场景进行拍摄，例如装备着 Xtion PRO 相机的移动机器人不断移动行走拍摄场景。由于从不同角度拍摄物体的图像数据的坐标是尚未被校正的，我们需要通过 OpenNI 的校正函数来修正 RGB 摄像头和 IR 摄像头产生的视差问题，使得深度图像和彩色图像重叠到相同位置上，这样方便了后面的纹理映射操作。效果图如下图 2-9 所示。

(a) RGB 图

(a) RGB image

(b) 深度图

(b) Depth image

图 2-9 图像数据

Fig.2-9 Image data

2.2.2 图像数据预处理

由于硬件制造工艺、物体表面反光系数差和环境因素等的影响，Xtion PRO 相机无法采集物体边缘或光滑表面的深度数据，获得的深度图像有一定的噪声和空洞，即是深度缺失块。在点云数据生成与配准之前，基于图像的空洞去噪操作能够很好的填补深度数据，有利于提取对应的图像特征。

本文采用采用最经典的双边滤波算法[45]进行平滑去噪。该算法是基于高斯滤波的改进算法，通过引入空间域权值和灰度域权值，计算待填补像素与相邻像素的空间距离和灰度相似性来得出待填补的像素值，这样有效地平滑图像并保持图像边缘。本文采用的双边滤波公式如下式(2.3)和式(2.4)所示，其公式原理是在一帧原始深度图像 O^{raw} 中取以 $p=(x,y)$ 为中心的矩形区域 θ，用矩形区域 θ 中的像素点 $q=(x',y')$ 的深度信息

来平滑图像，即是对原始深度图像进行平滑去噪处理得到新的深度图像。

$$\lambda(p,q)=\exp(-\frac{\|p-q\|^2}{2\delta_s^2}-\frac{\|O^{raw}(\mathrm{x})-O^{raw}(y)\|^2}{2\delta_r^2}) \tag{2.3}$$

$$O(p)=\frac{\sum_q[O^{raw}(x)\lambda(p,q)]}{\sum_q\lambda(p,q)} \tag{2.4}$$

其中，δ_s 是空间域权值，δ_r 是深度域权值。

2.3 点云数据

采集室内场景的三维信息是三维重建的首要元素，即是将实物模型转化成数字化模型。点云数据是指物体表面以点的形式呈现，每个点可以包括三维坐标信息(XYZ)、颜色信息(RGB)和激光反射强度。点云是指物体表面每个采样点的集合。

2.3.1 点云数据生成

Xtion PRO 相机采集的二维深度图像，包括了目标物与相机之间的距离信息以及目标物的二维像素坐标。根据二维深度图像，经过计算测量得到三维点云数据，并将二维像素坐标转化为世界坐标系的坐标数据。二维深度图像的像素点是由 16 位数据组成，其中高 13 位数据是深度值，即是相机与目标物的距离，低 3 位数据是 ID 号。经公式(2.5)处理，可得到每个像素点的深度值 $D(u)$ 。

$$D(u)=(\&0xfff8)>>3 \tag{2.5}$$

根据相机成像原理，相机的内参数矩阵为 K ，计算得到深度图像上的像素点 $u=(x,y)^T$ 对应的三维点坐标 $V(u)$：

$$V(u)=D(u)K^{-1}u \tag{2.6}$$

由于所得的三维点坐标是在相机坐标系下描述的，我们需要通过相机坐标到世界坐标的刚体变换式(2.7)，将三维点云转换到世界坐标系中，如式(2.8)；

$$T=\begin{bmatrix}R & t\\ 0^T & 1\end{bmatrix} \tag{2.7}$$

$$V^w(u)=TV(u) \tag{2.8}$$

2.3.2 点云数据去噪处理

Xtion PRO 相机采集的点云数据约有 30 万个像素点，体量庞大，且含有干扰的噪声点，因此需要有效的滤波方法对点云数据进行降噪处理。主要的滤波算法有：中值滤波，高斯滤波，拉普拉斯滤波和双边滤波[46]。其中，常用经典的双边滤波器处理深度图像中的噪声数据，但速度较慢。因此，为了实现更快对三维点云进行去噪处理，本文采用文献[47]提出的快速双边滤波器处理噪声，同时保留边缘信息。

2.3.3 点云配准与融合

通过 Xtion PRO 相机在不同视角下扫描采集得多帧点云数据，每帧点云数据之间存在重叠区域，而且都是基于各自相机坐标系的。我们就需要通过坐标变换将每帧点云数据转换到统一的世界坐标系下，这样便能实现不同视角下的点云数据的配准与融合。

本文只讨论对刚性物体的点云数据的配准，不考虑尺度变换，只考虑旋转变换和平移变换。在配准过程中，以当前帧点云数据 X^o 为初始值，使得经过坐标变换得到的 $(R*X^o+T)$ 与下一帧点云数据 X 之间的差值最小，即是使 $\|X-(R*X^o+t)\|^2$ 最小化求得旋转矩阵 R 和平移向量 t。点云配准流程图如下图 2-10 所示：

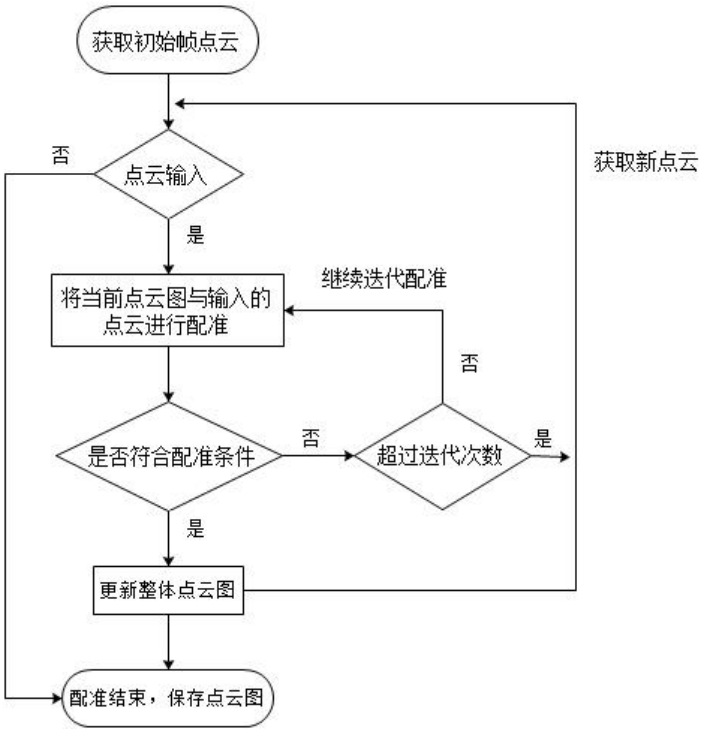


图 2-10 点云配准流程图

Fig.2-10 The flow diagram of point cloud matching

2.4 本章小结

本章主要介绍三维重建的硬件需求，三维点云数据的获取、处理以及配准融合的过程。先是详细介绍了 RGB-D 传感器的硬件结构，一般主流 RGB-D 传感器有微软 Kinect 相机和华硕 Xtion PRO 相机，解释了相机需要标定的原因，介绍了华硕 Xtion PRO 相机的标定方法及步骤。接着介绍了 Xtion PRO 相机如何获取 RGB 图像数据和深度图像数据，以及介绍通过双边滤波预处理对深度图像进行降噪和填补空洞。最后介绍了三维点云重建最重要的过程，即是如何生成点云数据，如何对点云数据进行去噪处理，如何实现点云数据的配准与融合。

第三章 视觉 SLAM 前端的实现

视觉 SLAM 前端是对获取的 RGB 图像进行提取特征点以及计算描述算子，利用特征描述算子对相邻两帧图像进行特征匹配，进而通过获取匹配特征点在对应的深度图像中的深度信息，得到相邻两帧匹配特征点对应的三维点坐标，根据匹配特征点与三维点的对应关系估计相机 6D 运动变换，并对局部相机姿态进行优化。视觉 SLAM 前端可分为特征提取与描述子计算、特征匹配、运动估计以及局部姿态优化这几个步骤。视觉 SLAM 前端流程图如下图 3-1 所示。本章将详细介绍视觉 SLAM 前端的各个步骤。

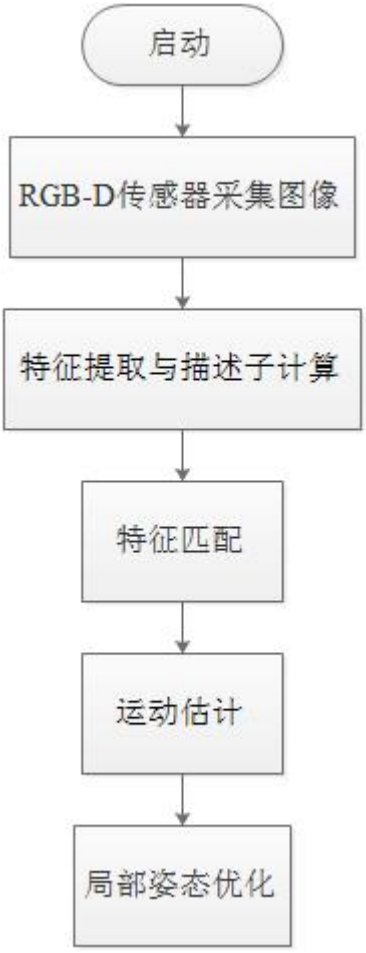


图 3-1 视觉 SLAM 前端流程图

Fig.3-1 The flow diagram of visual SLAM front-end

3.1 图像特征提取与匹配

RGB 图像特征点提取速度越快，更有利于实现实时的视觉 SLAM。特征匹配越精确，运动估计的累计误差越小，后续得到的二维和三维地图更接近真实场景。因此，图像特征提取与匹配是视觉 SLAM 中尤为关键的一步，后续的步骤都是基于图像特征提取与匹配这一步而工作。

3.1.1 特征提取算法分类

目前视觉 SLAM 中特征提取算法应用最为广泛有 SIFT[48]、SURF[49]和 ORB[50]。SIFT 算法提取的特征点数量多，误差小，但运算速度慢。SURF 算法提取的特征点数量较少，但是运算速度比 SIFT 算法快。而 ORB 是目前运算速度最快的一种特征提取算法。本节将详细介绍这三种特征提取算法。

(1)SIFT 算法

SIFT 全称为 Scale Invariant Feature Transform, 尺度不变特征变换。SIFT 算法是由 David Lowe[48]于 20 世纪末提出的一种特征提取算法，并于 2004 年完善总结。SIFT 算法检测与描述图像中的局部特征，能在空间尺度中找到极值点，进而提取出位置不变、尺度不变、旋转不变的特征点。SIFT 算法分为以下 4 个步骤：

1.尺度空间构建与极值检测

为了确保特征点的尺度不变性，使用尺度空间理论来模拟图像数据的多尺度特性。公式(3.1)定义了一帧二维图像的尺度空间，即是式(3.2)尺度可变高斯函数 $G(x,y,\sigma)$ 与图像 $I(x,y)$ 的卷积。

$$L(x,y,\sigma)=G(x,y,\sigma)*I(x,y) \tag{3.1}$$

$$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2} \tag{3.2}$$

其中 (x,y) 为图像坐标，σ 为尺度坐标，反映了图像的平滑程度。

利用不同尺度 σ 的高斯差分核与图像的卷积建立一个高斯差分尺度空间(DOG scale space)，如式(3.3)所示。

$$DoG(x,y,\sigma)=(G(x,y,k\sigma)-G(x,y,\sigma))*I(x,y)=L(x,y,k\sigma)-L(x,y,\sigma) \tag{3.3}$$

在查找高斯差分尺度空间的极值点时，我们将每个采样像素点与其相邻点进行比较，如果该点在高斯差分尺度空间的 26 个相邻点中是最大值点或最小值点，则认为该点为现尺度空间下的极值点。

2.关键点精确定位

通过拟合三维二次函数来精确确定关键点的位置和尺度，同时排除掉低对比度和在边缘附近难以定位的关键点，以提高抗噪声能力和匹配稳定性。

通过对高斯差分尺度 DoG 函数进行曲线拟合来提高关键点的稳定性。DoG 函数的泰勒展开式：

$$DoG(X) = DoG + \frac{\partial DoG^T}{\partial X} X + \frac{1}{2} X^T \frac{\partial^2 DoG}{\partial X^2} X \tag{3.4}$$

其中， $X = (x, y, \sigma)^T$

3.特征点分配指定方向

点 (x, y) 处的梯度模值和方向计算，如式(3.5)和(3.6)

$$m(x,y) = \sqrt{(L(x+1),y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{3.5}$$

$$\theta(x,y) = \alpha \tan 2((L(x,y+1) - L(x,y-1)) / (L(x+1,y) - L(x-1,y))) \tag{3.6}$$

其中 L 使用的尺度是特征点的尺度。目前到这一步为止，我们获得了特征点的位置、所处尺度以及方向三个信息，即是已经提取了图像的特征点。

4.特征点描述子计算

SIFT 描述子以较高的独特性来提高特征点正确匹配的概率，如图 3-2 所示，左图中央为特征点的中心点，选取 16*16 区域，再将其分成四个 4*4 小区域，每个小格代表特征点邻域所在尺度空间的一个像素，计算每个小区域的梯度直方图，每个直方图有 8 个方向，那么一共有 4*4*8=128 维向量，即是 SIFT 描述子向量。

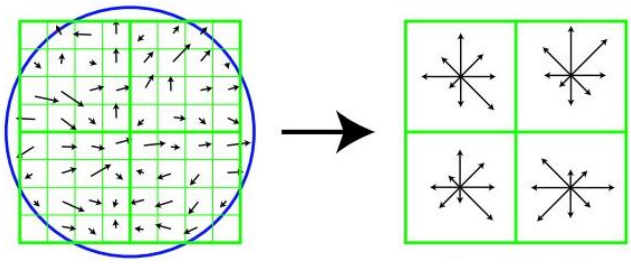


图 3-2 SIFT 特征点描述

Fig.3-2 SIFT feature point descriptors

通过上述内容可知，SIFT 可以获取丰富信息量，但计算复杂，计算量大，比较耗时，实时性比较差。

(2)SURF 算法

SURF 算法（Speeded Up Robust Features，加速鲁棒特征）算法是由 Herbert Bay 等

人[49]于2008年提出基于SIFT算法改进的一种算法，其运行效率比SIFT算法更快。SURF算法实现步骤如下：

1.特征点检测

SURF算法采用的是Hessian矩阵行列式近似值图像进行特征点检测。图像中一像素点$x=(u,v)$，则Hessian矩阵在x处尺度为δ的定义如式(3.7)。

$$H(x,\delta)=\begin{bmatrix} L_{xx}(x,\delta) & L_{xy}(x,\delta) \\ L_{xy}(x,\delta) & L_{yy}(x,\delta) \end{bmatrix} \tag{3.7}$$

使用框型滤波器来近似高斯二阶偏导数以提高卷积速度，经过滤波器后卷积D_{xx}、D_{yy}和D_{xy}分别近似$L_{xx}(x,\sigma)$、$L_{yy}(x,\sigma)$和$L_{xy}(x,\sigma)$。那么Hessian矩阵的行列式如式(3.8)所示。行列式的值大于阀值，则认为该点为特征点。

$$\det(H_{apporx})=D_{xx}D_{yy}-(0.9D_{yy})^2 \tag{3.8}$$

2.尺度空间构建

在这一步骤中，SURF算法与SIFT算法比较相似，有所不同的是SURF算法通过改变滤波器大小来构建尺度空间，允许同时处理尺度空间多层图像，提高了SURF算法运算速度。

3.特征点定位

在这一步骤中，SURF算法也是与SIFT算法比较相似，先是丢弃小于预设极值的取值，将经过Hessian矩阵处理过的每个点与其三维领域的26个点进行比较，如果它是最大值或者最小值，则认为该点为特征点。

4.特征点方向

SURF算法不统计梯度直方图，而是统计特征点领域内的Haar小波特征，这样能保证旋转不变性。以特征点为中心，计算其半径为6s(s为特征点所在的尺度值)的邻域内，统计60度扇形内所有点在水平和垂直方向的Haar小波响应总和，通过对响应值附加高斯权重系数，使得临近特征点的响应贡献大，将60度范围内的响应叠加编程新矢量，遍历整个圆形区域，选择最长矢量的方向为该特征点的主方向。

5.特征点描述子

在特征点周围构建一个有方向的方块，方块大小为20s*20s(s为特征点所在的尺度值)，其方向是上述步骤检测出来的主方向。把该方块分为16个子方块，统计每个子方块中25个像素的水平方向和垂直方向的haar小波特征响应值，进而得到特征点描述

子。

(3)ORB 算法

ORB(Oriented FAST and Rotated BRIEF)算法是由 Ethan 等人[50]于 2011 年基于结合 BRIEF(Binary Robust Independent Elementary Features)特征描述算法[51]和 FAST(Features from Accelerated Segment Test)角点提取算法[52]的优点提出了一种性能优越的二进制特征提取算法。BRIEF 算法的优点在于提取特征点速度极快，但该算法不具备旋转不变性，对噪声敏感以及不具备尺度不变性，因此 ORB 算法引入 FAST 算法思想来弥补 BRIEF 算法的不足。

1.特征点提取

FAST 算法通过检测局部图像中是否存在角点，无需计算二阶导数，也就省略了去噪声的步骤，从而提高了运算效率。如图 3-3 所示，点 p 是待检测点，计算点 p 周围的 16 个点与点 p 灰度值的差值是否足够大，如果 16 个灰度差值大于某个阈值的个数足够多，则认为点 p 为一个特征点，如式(3.9)所示。

$$N=\sum_{1\le i\le 16}|I(x_i)-I(p)|>t \tag{3.9}$$

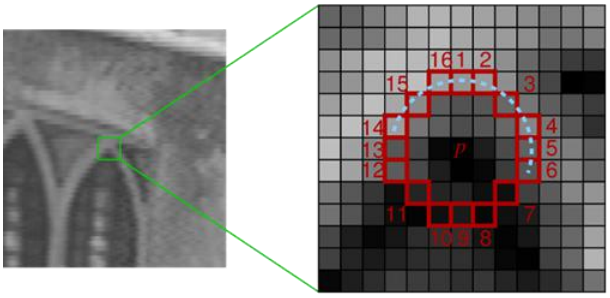


图 3-3 角点检测

Fig.3-3 Corner detection

满足判定条件的像素点个数 N，$I(p)$ 为待检测点 p 的灰度值，$I(x_i)$ 为圆周上像素点的灰度值，t 为设定的阈值，N 一般为 12 或 9。为了提高检测效率，先检查点 p 垂直水平方向上的四点（即图中 1,5,9,13），这 4 个点中至少有 3 个点与候选点灰度值的差值足够大，否则不用计算其它点，认为该候选点不是特征点。

2.特征点描述子

ORB 算法选择了 BRIEF 特征描述，其优点在于特征描述算子结构简单，能够更快地完成匹配，但其缺点是不具备旋转不变性。因此 ORB 算法使用了主方向来引导 BRIEF，即是 steered BRIEF 方法。任意一个特征点的 BRIEF 描述子是一个长度为 n 的

二值码串，二值码串是由特征点邻域n个点对（即$2n$个点）组成，其矩阵S：

$$S=\begin{pmatrix} x_1 & x_2 & \cdots & x_{2n} \\ y_1 & y_2 & \cdots & y_{2n} \end{pmatrix} \tag{3.10}$$

ORB 算法在特征点提取时已经确定了特征点的方向角θ。利用θ和其对应的旋转矩阵R_θ来矫正S，为S_θ。

$$S_\theta = R_\theta S \tag{3.11}$$

$$\text{其中，}\ R_\theta=\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{3.12}$$

即可得旋转不变的特征描述子：

$$g_n(I,\theta) \coloneqq f_{nd}(I) \,|\, (x_i, y_i) \in S_\theta \tag{3.13}$$

综合上述关于 SIFT 算法、SURF 算法以及 ORB 算法的描述，ORB 算法提取特征点以及计算描述子的运算速度最快。因此，为了实现实时的视觉 SLAM，本文在特征点提取和特征描述子计算方面采用 ORB 算法。

3.1.2 特征匹配

匹配确定两帧图像中特征点的对应关系是计算机视觉研究领域使用最多的处理方法，例如有应用在图像拼接，图像识别，目标跟踪等诸多领域。实现视觉 SLAM 过程中，为了更优的相机位姿估计、关键帧选取以及闭环检测，需要对特征点描述子进行特征匹配与跟踪。实现图像特征点的匹配必须解决三个关键问题：一是特征描述子向量的相似性度量；二是两个特征点是匹配的判定条件；三是如何剔除误匹配[53]。

在实际计算中，我们需要计算两个描述子间的距离的度量准则来判断相似性。目前惯用的距离计算方法有四种：欧氏距离（Euclidean distance），马氏距离（Mahalanobis distance），汉明距离（Hamming distance）以及海宁格距离（Hellinger distance）。

ORB 的二值字符串描述子是采用汉明空间表示的，因而挑选最小汉明距离为相似对。两帧图像的特征点描述子为：

$$K_1 = x_0 x_1 ... x_n \,, K_2 = y_0 y_1 ... y_n \tag{3.14}$$

计算汉明距离之间的异或之和表示两个 ORB 特征点描述子的相似度，如式(3.15)所示。

$$D(K_1, K_2) = \sum_{i=0}^{n} x_i \otimes y_i \tag{3.15}$$

其中，$D(K_1,K_2)$ 越小代表相似度越高。由于汉明距离只需要在相同位求异或操作，可见其复杂度远低于欧式距离的求法。本文已经采用了 ORB 算法进行特征点提取和特征描述子计算，而且 ORB 特征二值字符串描述子在匹配策略上很有优势，最后采用 RANSAC（Random Sample Consensus，随机采样一致性）算法[54]进行去除误匹配。

3.2 关键帧选取

关键帧的选取是视觉 SLAM 中一个非常关键的步骤。因为在视觉 SLAM 系统运行时，华硕 Xtion PRO 以 30fps 的频率获取 RGB 图像和深度图像，如果将获取的每一帧图像都视为关键帧，即是需要处理图像的数量庞大，过多的关键帧会导致三维重建时点云更新过于密集频繁，极大增加运算量和耗费内存资源，而且造成后端图优化的节点数量剧增，这就要求更高的算法。移动机器人行走过程中捕获的连续图像序列中，相邻两帧图像间距往往非常小，甚至是同一场景的成像，这样图像之间相似性极高，因此将所有图像都视为关键帧并且计算处理每一帧是毫无意义的做法。

还有另外一种情况，如果选取关键帧的数量过少，会错误判断移动机器人运动过大，导致相邻帧图像间的重叠区域过小，从而引起图像特征点匹配失败，便视觉 SLAM 过程中容易出现运动跟踪丢失现象。因此可知准确地从 Xtion PRO 捕获图像序列中选取关键帧是至关重要的。

本文通过计算位移与旋转的范数加和来度量运动大小，如式(3.16)所示。其中，e 为运动大小，Δt 为平移大小，r 为旋转大小。

$$e=\|\Delta t\|+\min(2\pi-\|r\|,\|r\|) \tag{3.16}$$

设关键帧序列为 F，运动最大阀值为 E_{error}，运动最小阀值 E_{key}，关键帧选取流程如下：

1.初始化关键帧序列 F，将第一帧图像 I_0 存入 F。

2.捕获一帧新图像 I，通过式(3.16)计算关键帧序列 F 中最后一帧图像 I_n 与 I 之间的运动大小 e。

3.如果 $e>E_{error}$，表明运动过大，也可能是计算错误的原因，因而丢弃该帧图像。

4.如果没有匹配上或者匹配数量过少，表明该帧图像质量不高，因而丢弃该帧图像。

5.如果 $e<E_{key}$，表明该帧图像与前一关键帧距离过近，因而丢弃该帧。

6.将该帧新图像 I 视为新关键帧，并存入到关键帧序列 F 中。

3.3 运动估计方法

通过上述的特征点提取与匹配环节，已经确定了两帧图像间相同特征点的一一对应关系。本节中，我们将进一步根据图像间特征点对应关系来求解移动机器人运动，即是运动估计。

3.3.1 三维匹配点

RGB 图像中点 (u,v) 对应深度图像的深度值为 $dep(u,v)$，可通过式(3.17)求得 (u,v) 对应的三维坐标点 (x,y,z)。

$$\begin{cases} x = \dfrac{(u-c_x)z}{f_x} \\ y = \dfrac{(v-c_y)z}{f_y} \\ z = dep(u,v) \end{cases} \tag{3.17}$$

其中，f_x, f_y 为相机的焦距，c_x, c_y 为相机的光圈中心，均为已知。

通过特征点提取与匹配得到二维匹配点对集，利用式(3.17)求得每个特征点在空间中的三维坐标值，便组成一系列三维匹配点对集。

3.3.2 初步运动估计

获得两帧图像间的三维匹配点对集之后，根据三维匹配点之间的相对位姿变化关系来估计出一个较优的运动模型。本文采用 RANSAC 算法来求解能满足整个匹配点集的最优运动参数，即是旋转矩阵 R 和平移向量 t。

RANSAC 算法需要设置迭代次数与阀值大小，其中根据所期望获得的位姿转换为最优位姿转换的概率来设定迭代次数，所处理的问题以及经验来设定阀值大小。随机选取 n 个特征点，p 为多次迭代中至少有一次获得采样点集不包括局外点的概率，x 为一次采样获得局内点的概率，$y = 1 - x$ 为一次采样获得局外点的概率，N 为迭代次数，这样可令等式(3.18)成立。

$$\begin{aligned} 1 - p &= (1 - x^n)^N \\ N &= \frac{\log(1-p)}{\log(1-(1-y)^n)} \end{aligned} \tag{3.18}$$

RANSAC 过程求解出的位姿转换 $T_{[R|t]}$ 是由一个 3×3 矩阵的旋转分量 R 和一个 3×1

向量的平移分量t，通过式(3.19)可以求得已知三维点P经过位姿转换$T_{[R|t]}$之后的投影$P^{'}$。

$$P^{'}=T_{[R|t]}P \tag{3.19}$$

3.3.3 优化位姿转换

本文使用 ICP（Iterative Closest Point，迭代最近点）算法[55]进行优化位姿转换，其使用迭代方式求解两个匹配点集之间的刚体变换，便使转换后点集间距离最小。

利用 ICP 算法对两个具有运动关系的二维特征点集以及对应的三维点集进行配准对齐，以计算出点集间的旋转和平移转换关系，即是求解旋转矩阵R和平移向量t。先是通过图像匹配确定了特征点的对应关系，这样大大减少错误点的数量，确保 ICP 匹配时收敛于最优值。特征点提取与匹配后的两帧图像，结合对应的深度信息图像分别得到各自的特征点三维点集，分别是$P=\{p_i \mid i=1,...,n\}$和$Q=\{q_i \mid i=1,...,n\}$，每个三维点坐标为$(x,y,z)_i^T$形式。求解式(3.20)这个最小二乘问题的最优解就是 ICP 算法计算运动估计的过程，即是连续迭代得到从点集P到点集Q（即是两帧图像间）的旋转矩阵R和平移向量t。

$$\min E=\sum_{i=1}^{n}(q_i-(Rp_i+t))^2 \tag{3.20}$$

由于图像特征点匹配时依然可能会存在错误匹配，一些特征点对应关系错误，以及特征点深度信息也会有误差等原因，导致 ICP 配准图像特征点三维点集时也会存在噪点干扰问题。因此本文先是初步使用 RANSAC 算法来剔除那些误差大的噪点，为了达到更加稳定准确的运动估计。大致流程：先是从匹配点集中随机选取 3 对匹配点，如果三对匹配点不符合 ICP 算法的基本条件，则重新选取，满足条件下利用 ICP 算法估计这三对匹配点的旋转向量R和平移向量t，再根据式(3.21)来累计内点数目，重复随机取匹配点到累计内点数目的过程，迭代 N 次后将内点数目最多的旋转矩阵R和平移向量t视为运动估计。

$$(q_i-(Rp_i+t))^2<\varepsilon \tag{3.21}$$

其中，ε为内点判定阀值。

3.4 本章小结

本章首先是简要介绍视觉 SLAM 前端的实现流程，详细介绍了 SIFT 算法、SURF 算法和 ORB 算法的原理，本文采用 ORB 算法来提高在特征点提取和特征描述子计算速度，极大提高了视觉 SLAM 的实时性。然后介绍了特征点匹配的原理过程，并详细描述了如何利用最小汉明距离与 RANSAC 算法剔除错误 ORB 特征点匹配。接着为了降低运算量和内存资源，又防止相机运动跟踪丢失，本章讲述了关键帧选取的重要性，介绍了选取关键帧的方法以及流程。最后，讨论了根据图像间特征点对应的三维匹配点对集，利用 RANSAC 算法初步求解出相机运动估计，进而利用 ICP 算法优化位姿转换关系。

第四章 视觉 SLAM 后端的实现

视觉 SLAM 后端大致分为两类：基于滤波器的视觉 SLAM 和基于图优化的视觉 SLAM。基于滤波器的视觉 SLAM 是采用递归贝叶斯方法，推算出部分未知的非结构化环境。常见的基于滤波器的视觉 SLAM 分两种：卡尔曼滤波器(KF)及其相关的改进算法、粒子滤波器(PF)算法。基于滤波器的视觉 SLAM 无法重线性化和秩增加，而且要求系统计算能够快速完成线性化和更高的更新效率，因而其很难被用于大型环境的地图建立。由于有研究者发现 SLAM 的稀疏性质，利用图优化方法求解视觉 SLAM 问题逐渐成熟，图优化也已经成为求解 SLAM 问题的主流方法。图优化方法与闭环检测方法有机结合起来，能够实现视觉 SLAM 在大规模环境场景下建立地图与定位。

4.1 基于卡尔曼滤波的 SLAM 后端

数学家 Rudolf Emil Kalman[56]于 20 世纪 50 年代首次提出了卡尔曼滤波 KF 的思想及理论。卡尔曼滤波是一种贝叶斯滤波，可以通过数量不多的参数来表征单峰多元分布，这实质上是一种基于自回归思想的最优化数据处理方法，非常适用于解决大部分效率问题。卡尔曼滤波有几种改进的变型：相关信息滤波(Information Filter，IF)、扩展卡尔曼滤波(Extend Kalman Filter，EKF)和扩展信息滤波 IF(Extend Information Filter，EIF)。目前卡尔曼滤波主要应用在机器人导航规划、IMU 数据融合以及图像处理如人脸识别、图像分割等领域。

基于卡尔曼滤波的 SLAM 假设状态变换和测量函数式是线性的加性高斯噪声，以及初始的后验概率也是高斯分布。对卡尔曼滤波 SLAM 的观测方程和运动方程作了恰当的假设，是被视为一个高斯噪声融合的过程。

基于卡尔曼滤波的 SLAM 的运动方程描述的是里程计传感器工作的过程，如式(4.1)所示。

$$x_i = f_i(x_{i-1}, u_i) + w_i \tag{4.1}$$

其中，$f_i(x_{i-1}, u_i)$代表运动模型，w_i是测量过程中引入的呈正态分布的零均值噪声，x_i是机器人运动轨迹。

基于卡尔曼滤波的 SLAM 的观测方程如下式(4.2)：

$$z_k = h_k(x_{ik}, l_{jk}) + v_k \tag{4.2}$$

其中，$h_k(x_{ik}, l_{jk})$代表观测模型，v_k是呈正态分布的零均值测量噪声。

那么，我们可以使用式(4.3)表示基于卡尔曼滤波的 SLAM 问题。此方法的时间更新公式如式(4.4)所示。

$$\begin{aligned} [x_{i|i}, m_i] &= E[x_i, m_i \mid Z_{1:i}] \\ P_{i|j} &= \begin{bmatrix} P_{xx} & P_{xm} \\ P_{xm}^T & P_{mm} \end{bmatrix} \end{aligned} \tag{4.3}$$

$$\begin{aligned} x_{i|i-1} &= f(x_{i-1|i-1}, u_i) \\ P_{xx,i|i-1} &= \Delta f P_{xx,i-1|i-1} \Delta f + R_i \end{aligned} \tag{4.4}$$

观测更新如下式(4.5)所示：

$$\begin{aligned} P_i &= (I - K_i H_i) P_{i|i-1} \\ K_i &= P_{i|i-1} H_i^T (H_i P_{i|i-1} H_i^T + P_R) \end{aligned} \tag{4.5}$$

其中，P为协方差矩阵，K为卡尔曼增益，H为测量模型雅可比矩阵。

4.2 基于粒子滤波的 SLAM 后端

粒子滤波是以递推贝叶斯估计为基础的蒙特卡洛滤波方法(Monte Carlo Methods)，粒子滤波器的核心思想是通过从后验概率中抽取的随机状态粒子来表达其分布，先是依据已知先验信息求解状态变量的建议分布，再从建议分布的空间中生成一堆粒子，接着在递推迭代估计过程中逐渐调整粒子权值，其中这些粒子权值用于表示状态变量的概率分布。粒子滤波是用一组滤波器来估计机器人的位姿，每个滤波器对应着一个路标位置，通过观测对每个路标位置进行加权传递，使用概率密度表示机器人的位置，其中概率最高的粒子表示机器人当前的实际位姿。

基于粒子滤波的 SLAM 的特点是使用粒子表示非线性和非高斯分布的位姿分布，这一特点是与基于卡尔曼滤波的 SLAM 方法的最大区别。假设已知机器人的路径估计，这样位置路标估计之间是独立的，对 SLAM 问题中后验概率分布$p(x_{1:k}, m \mid z_{1:k}, u_{0:k-1}, x_0)$进行分解，如下式(4.6)所示：

$$\begin{aligned} &p(x_{1:k}, m \mid z_{1:k}, u_{0:k-1}, x_0) = p(m \mid x_{0:k}, z_{1:k}, u_{0:k-1}) p(x_{1:k} \mid z_{1:k}, u_{0:k-1}, x_0) \\ &= p(x_{1:k} \mid z_{1:k}, u_{0:k-1}, x_0) \prod_{i=1}^{M} p(m_i \mid x_{1:k}, z_{1:k}) \end{aligned} \tag{4.6}$$

设粒子集为$\left\{x_{1:k}^{'}, w_k^{'}\right\}_{i=1}^{N}$，$k$时刻的控制输入$u_k$和$k+1$时刻的观测值$z_{k+1}$，基于粒子滤

波的 SLAM 的实现主要步骤：

1.初始化

明确移动机器人姿态初始值 $\hat{x}_0$，协方差矩阵 P_0，整个过程噪声协方若矩阵 Q_0，测量噪声协方差矩阵 R_0 等相关参数，采集初始中的 N 个样本 $\{x_0^{'}, w_0^{'}\}\, i=1,2,...,N$。

2.预测

往机器人的运动模型输入预测机器人 $k+1$ 时刻的位姿，即是每个粒子在 $k+1$ 时刻位姿的预测均值和方差。

3.数据关联

将记录 $k+1$ 时刻的观测值，与 k 时刻每个粒子的估计观测值进行数据关联，每个粒子的关联过程是相互独立的。

4.获取建议分布

利用 EKF 算法[57]基于每个粒子各自的观测值，计算每个粒子位姿估计的均值和方差，求解重要性概率密度函数，该概率密度函数即是每个粒子的建议分布 $q(x_{k+1} \mid x_k^i, z_{k+1}, u_k)$。

5.机器人路径估计

利用 RBPF 算法对机器人路径进行估计，计算 $k+1$ 时刻机器人后验概率分布 $p(x_{1:k+1} \mid z_{1:k+1}, u_{0:k}, x_0)$ 的粒子集 $\{x_{1:k+1}^i, w_{k+1}^i\}_{i=1}^N$。

6.更新地图估计

场景地图的估计已被分解成 M 个相互独立的特征估计问题，采用 EKF 算法进行估计地图特征的后验概率分布 $p(m_i \mid x_{1:k+1}, z_{1:k+1}, u_{1:k+1})$。地图估计表示：

$$\{\mu^1, P^1, ..., \mu^M, P^M\} \tag{4.7}$$

式中，μ^i 表示场景地图第 i 个特征的高斯均值，P^i 表示场景地图中第 i 个特征方差。

7.当时间未停止时，则重回步骤 2 继续执行。

4.3 基于图优化的 SLAM 后端

图优化是一种非线性化与图论相结合的理论。基于图优化的 SLAM 利用位姿图(Pose Graph)来表示移动机器人的运动轨迹和场景地图，通过对全局环境信息数据进行处理，利用边缘化的方法将场景特征转换成位姿间的约束，进而估计完整位姿序列，

而不像基于滤波 SLAM 那样只计算机器人当前位姿。如图 4-1 所示，图是由节点和边构成，在基于图优化的 SLAM 中，优化图中的节点(node)或顶点(vertex)表示机器人不同时刻的位姿，边(edge)表示不同时刻位姿之间的约束关系，例如$t+1$时刻和t时刻之间的里程计关系构成边，或通过视觉图像计算得到的位姿转换矩阵构成边。基于图优化的 SLAM 后端的目标是调整优化图中机器人位姿节点所处的位置，使其尽量满足边所表示的约束关系，获得一张最优位姿图。

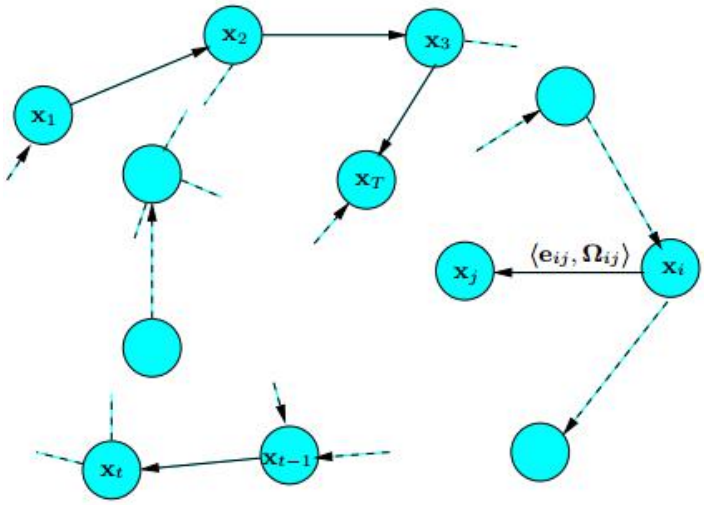


图 4-1 机器人位姿作为节点的优化图

Fig.4-1 Optimization graph that robot pose as a node

4.3.1 视觉 SLAM 图优化模型构建

设 $\mathbf{x}$ 为机器人的位姿，即是位置坐标和姿态，T 为不同时刻机器人位姿之间的变换关系，而 $\mathbf{x}$ 与 T 是一个四阶方阵表示的姿态矩阵，如下式(4.8):

$$\mathbf{x}, \mathrm{T} \overset{def}{=} \begin{bmatrix} R_{3\times3} & t_{3\times1} \\ 0 & 1 \end{bmatrix} \tag{4.8}$$

其中，R 为旋转矩阵，t 为平移向量。

假设第 i 帧图像和第 j 帧图像对应的移动机器人位姿 $\mathbf{x}_i$ 和 $\mathbf{x}_j$，求解两者之间的真实位姿变换关系 $\hat{\mathrm{T}}_{i,j}$，位姿 $\mathbf{x}_j$ 在 $\mathbf{x}_i$ 坐标系下可表示为：

$$\mathbf{x}_i = \hat{\mathrm{T}}_{i,j}\, \mathbf{x}_j \tag{4.9}$$

通过根据第 i 帧图像和第 j 帧图像之间的运动估计可以获得对应机器人的观测位姿变换关系 $\mathrm{T}_{i,j}$，在理想情况下计算求解的真实值 $\hat{\mathrm{T}}_{i,j}$ 与位姿估计值 $\mathrm{T}_{i,j}$ 是完全一致的，但

实际上由于噪声等因素的影响，导致较大误差，利用误差函数 $e_{i,j}$ 来表示该差值，如式(4.10)所示。

$$e_{i,j}(\mathbf{x_i},\mathbf{x_j}) = \mathrm{T}_{i,j} - \hat{\mathrm{T}}_{i,j}(\mathbf{x_i},\mathbf{x_j}) \tag{4.10}$$

因为图像帧间匹配运动估计会产生该误差 $e_{i,j}$，而我们的目的是使误差总和最小化，进而使机器人的整个运动轨迹更加接近真实值，其实误差最小化问题即是求解最小二乘优化问题：

$$F(\mathbf{x}) = \sum_{\langle \mathbf{i},\mathbf{j}\rangle \in C} e(\mathbf{x_i},\mathbf{x_j},\mathbf{z_{ij}})^T \Omega_{i,j} e(\mathbf{x_i},\mathbf{x_j},\mathbf{z_{ij}}) \tag{4.11}$$

其中，$\mathbf{x_i}$，$\mathbf{x_j}$ 表示机器人位姿，$\mathbf{z_{ij}}$ 表示测量值，$\Omega_{i,j}$ 为帧间误差信息矩阵。$e(\mathbf{x_i},\mathbf{x_j},\mathbf{z_{ij}})$ 是一个向量误差函数，表示 $\mathbf{x_i}$ 和 $\mathbf{x_j}$ 之间的关系与测量值 $\mathbf{z_{ij}}$ 之间吻合度，将该误差函数 $e(\mathbf{x_i},\mathbf{x_j},\mathbf{z_{ij}})$ 简化为 $e_{\mathbf{ij}}(\mathbf{x})$。

找到机器人位姿集 $\mathbf{x}^*$ 可使 $F(x)$ 最小，如式(4.12)，那么该位姿集 $\mathbf{x}^*$ 为机器人轨迹最优值。

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} F(\mathbf{x}) \tag{4.12}$$

4.3.2 图优化的求解推导过程

图优化问题就是求解最小二乘问题，但是机器人位姿之间的变化函数是非线性的，即是个非线性最小二乘问题，可利用迭代法进行求解，例如 Guass-Newton 法或 Levenberg-Marquardt 法。

假设初始值 $\check{\mathbf{x}}$，将误差函数 $e_{\mathbf{ij}}(\mathbf{x})$ 在该初始值附近进行一阶泰勒展开，迭代求得最优解，如式(4.13)。

$$e_{\mathbf{ij}}\left(\check{\mathbf{x}}_\mathbf{i}+\Delta\mathbf{x}_\mathbf{i},\check{\mathbf{x}}_\mathbf{j}+\Delta\check{\mathbf{x}}_\mathbf{j}\right) = e_{\mathbf{ij}}\left(\check{\mathbf{x}}+\Delta\mathbf{x}\right) \cong e_{\mathbf{ij}} + \mathbf{J}_{\mathbf{ij}}\Delta\mathbf{x} \tag{4.13}$$

其中，$\mathbf{J}_{\mathbf{ij}}$ 为误差函数 $e_{\mathbf{ij}}(\mathbf{x})$ 在初始值 $\check{\mathbf{x}}$ 附近的雅克比矩阵，$e_{\mathbf{ij}}$ 代替 $e_{\mathbf{ij}}(\check{\mathbf{x}})$。

将上式(4.13)代入误差平方和(4.11)中，可得：

$$
\begin{aligned}
&F_{\mathbf{ij}}\left(\breve{\mathbf{x}}+\Delta\mathbf{x}\right)\\
&=e_{\mathbf{ij}}\left(\breve{\mathbf{x}}+\Delta\mathbf{x}\right)^{T}\Omega_{\mathbf{ij}}e_{\mathbf{ij}}\left(\breve{\mathbf{x}}+\Delta\mathbf{x}\right)\\
&\cong\left(e_{\mathbf{ij}}+\mathbf{J}_{\mathbf{ij}}\Delta\mathbf{x}\right)^{T}\Omega_{\mathbf{ij}}\left(e_{\mathbf{ij}}+\mathbf{J}_{\mathbf{ij}}\Delta\mathbf{x}\right)\\
&=\underbrace{e_{\mathbf{ij}}^{T}\Omega_{\mathbf{ij}}e_{\mathbf{ij}}}_{c_{\mathbf{ij}}}+2\underbrace{e_{\mathbf{ij}}^{T}\Omega_{\mathbf{ij}}\mathbf{J}_{\mathbf{ij}}}_{b_{\mathbf{ij}}^{T}}\Delta\mathbf{x}+\Delta\mathbf{x}^{\mathbf{T}}\underbrace{\mathbf{J}_{\mathbf{ij}}\Omega_{\mathbf{ij}}\mathbf{J}_{\mathbf{ij}}}_{H_{\mathbf{ij}}}\Delta\mathbf{x}\\
&=c_{\mathbf{ij}}+2b_{\mathbf{ij}}^{T}\Delta\mathbf{x}+\Delta\mathbf{x}^{\mathbf{T}}\mathbf{H}_{\mathbf{ij}}\Delta\mathbf{x}
\end{aligned}
\tag{4.14}
$$

其中，$b_{\mathbf{ij}}$，$\Delta\mathbf{x}$和$e_{\mathbf{ij}}$是列向量，$c_{\mathbf{ij}}$是一个数值。

$$
\begin{aligned}
&F\left(\breve{\mathbf{x}}+\Delta\mathbf{x}\right)\\
&=\sum_{\langle\mathbf{i},\mathbf{j}\rangle\in C}F_{\mathbf{ij}}\left(\breve{\mathbf{x}}+\Delta\mathbf{x}\right)\\
&\cong\sum_{\langle\mathbf{i},\mathbf{j}\rangle\in C}c_{\mathbf{ij}}+2b_{\mathbf{ij}}\Delta\mathbf{x}+\Delta\mathbf{x}^{T}\mathbf{H}_{\mathbf{ij}}\Delta\mathbf{x}\\
&=c+2b^{T}\Delta\mathbf{x}+\Delta\mathbf{x}^{T}\mathbf{H}\Delta\mathbf{x}
\end{aligned}
\tag{4.15}
$$

其中，$c=\sum c_{\mathbf{ij}}$，$b=\sum b_{\mathbf{ij}}$，$\mathbf{H}=\sum\mathbf{H}_{\mathbf{ij}}$。求解(4.15)式使得误差平方和函数$F\left(\breve{\mathbf{x}}+\Delta\mathbf{x}\right)$最小，即是求其一阶导数并使其等于 0，如下式(4.16)。

$$
\mathbf{H}\Delta\mathbf{x}^{*}=\text{-}b \tag{4.16}
$$

$\mathbf{H}$是基于图优化 SLAM 的信息矩阵，也是由每个节点间的误差函数的雅克比矩阵$\mathbf{J}_{\mathbf{ij}}$叠加得到。用于机器人位姿观测的地标分布的分散性，使得基于图优化 SLAM 具有稀疏性，进而导致$\mathbf{H}$矩阵的稀疏性，即雅克比矩阵$\mathbf{J}_{\mathbf{ij}}$和$\mathbf{x}_{\mathbf{i}}$、$\mathbf{x}_{\mathbf{j}}$无关的列都为 0。那么我们可以通过 Cholesky 分解来解式(4.16)所示的矩阵方程，则机器人的位姿为初始估计值加上求得的增量：

$$
\mathbf{x}^{*}=\mathbf{x}+\Delta\mathbf{x}^{*} \tag{4.17}
$$

由式(4.16)可知，首先需要计算系统信息矩阵$\mathbf{H}$和b向量，才能求解出增量$\Delta\mathbf{x}^{*}$。由式(4.15)可知，每个位姿间的约束边影响着系统信息矩阵$\mathbf{H}$和b向量，在$\mathbf{H}$和b中添加部分的位置取决于位姿约束误差函数的雅克比矩阵，其中误差函数的雅克比矩阵如下式(4.18)：

$$
\mathbf{J}_{\mathbf{ij}}=\left(0\cdots0\underbrace{\frac{\partial e_{\mathbf{ij}}\left(\mathbf{x}_{\mathbf{i}},\mathbf{x}_{\mathbf{j}}\right)}{\partial\mathbf{x}_{\mathbf{i}}}}_{\mathbf{A}_{\mathbf{ij}}}0\cdots0\underbrace{\frac{\partial e_{\mathbf{ij}}\left(\mathbf{x}_{\mathbf{i}},\mathbf{x}_{\mathbf{j}}\right)}{\partial\mathbf{x}_{\mathbf{j}}}}_{\mathbf{B}_{\mathbf{ij}}}0\cdots0\right) \tag{4.18}
$$

其中， $\mathbf{A}_{i,j}$ 为误差函数 $e_{i,j}(\mathbf{x}_i,\mathbf{x}_j)$ 对 $\mathbf{x}_i$ 的偏导， $\mathbf{B}_{i,j}$ 为误差函数 $e_{i,j(\mathbf{x}_i,\mathbf{x}_j)}$ 对 $\mathbf{x}_j$ 的偏导。

由式(4.14)可知：

$$\begin{aligned}\mathbf{H}_{ij} &= \mathbf{J}_{ij}^T\Omega_{ij}\mathbf{J}_{ij}\\ &= \begin{pmatrix}\vdots\\ \mathbf{A}_{ij}^T\\ \vdots\\ \mathbf{B}_{ij}^T\\ \vdots\end{pmatrix}\Omega_{ij}\begin{pmatrix}\cdots & \mathbf{A}_{ij} & \cdots & \mathbf{B}_{ij} & \cdots\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{A}_{ij}^T\Omega_{ij}\mathbf{A}_{ij} & \mathbf{A}_{ij}^T\Omega_{ij}\mathbf{B}_{ij}\\ \mathbf{B}_{ij}^T\Omega_{ij}\mathbf{A}_{ij} & \mathbf{B}_{ij}^T\Omega_{ij}\mathbf{B}_{ij}\end{pmatrix}\end{aligned} \tag{4.19}$$

也可求得：

$$\begin{aligned}b_{ij}^T &= e_{ij}^T\Omega_{ij}\mathbf{J}_{ij}\\ &= e_{ij}^T\Omega_{ij}\begin{pmatrix}0 & \cdots & \mathbf{A}_{ij} & \cdots & \mathbf{B}_{ij} & \cdots & 0\end{pmatrix}\\ &= \begin{pmatrix}0 & \cdots & e_{ij}^T\Omega_{ij}\mathbf{A}_{ij} & \cdots & e_{ij}^T\Omega_{ij}\mathbf{B}_{ij} & \cdots & 0\end{pmatrix}\end{aligned} \tag{4.20}$$

即是：

$$b_{ij} = \begin{pmatrix}\vdots\\ \mathbf{A}_{ij}^T\Omega_{ij}e_{ij}\\ \vdots\\ \mathbf{B}_{ij}^T\Omega_{ij}e_{ij}\\ \vdots\end{pmatrix} \tag{4.21}$$

将每个位姿约束的误差函数求得 $\mathbf{H}_{ij}$ 和 b_{ij} 累积可得到 $\mathbf{H}=\sum\mathbf{H}_{ij}$ 和 $b=\sum b_{ij}$，然后通过式(4.16)即可求得 $\Delta\mathbf{x}^*$。

4.3.3 帧间约束信息矩阵的确定

基于图优化的 SLAM 是通过帧间特征匹配与运动估计来构建每帧图像间的约束关系，因为移动机器人运动距离大小不同以及每帧图像视角也不一样，所以每对图像间的运动估计的误差更是不同。例如当相机捕获相邻两帧图像的共同特征点比较多，以及特征点的深度值较小时，相机运动估计的精确性和可靠性都挺高的。当相机拍摄纹理不明显的地板或者光滑纯色的墙壁等场景，因特征点过少导致运动估计误差过大。机器人位姿间运动估计的误差不一致体现了优化图中位姿节点间边的约束力不一致，

将边的约束力不一致形容为阻尼系数不同的弹簧。从上一部分的图优化的求解推导过程可知，帧间误差信息矩阵Ω_{ij}表示的是位姿节点间边的约束力的大小。图优化模型中，每一条边代表一个测量值，每一条边的信息矩阵即是对应边的测量协方差矩阵的逆，协方差越小表示位姿间的运动估计误差越小，精度越高，也即是信息矩阵相应的值更大。

依据图像帧间运动估计的精确度来确定每条约束边的信息矩阵，运动估计精确性判断标准有两个：一方面是计算运动估计中 RANSAC 三维配准后的内点数量$n_{inliers}$，内点数量$n_{inliers}$越大，说明两帧图像间的共同特征点越多，两帧图像公共视野更多，运动估计的误差也更小。另一方面，运动估计的精确度与内点数量占匹配对的比率有关。先通过图像间的特征匹配确定图像间的特征点对应关系，若图像特征匹配对数量大于规定最小阀值，则利用图像二维特征点对应的三维点信息进行空间三维 RANSAC 匹配求解得图像间的变换矩阵关系。图像特征匹配对数量足够以及经过 RANSAC 三维匹配后大部分三维点为合格内点，说明图像间的变换估计可靠高，精度较高。但当内点数量占匹配对比率不高时，则说明图像间运动估计稳定性差，误差可能较大。利用式(4.22)计算内点数量占匹配对的比率$p_{inliers}$：

$$p_{inliers}=\frac{n_{inliers}}{n_{matchpoints}}\times 100\% \tag{4.22}$$

其中，$n_{matchpoints}$为图像特征匹配对数量，$n_{inliers}$为三维匹配内点数量。

那么，可以利用式(4.23)计算帧间运动估计精确度：

$$\sigma=0.05+0.1*\exp\left\{\frac{(n_{inliers}-n_{threshold})*p_{inliers}}{10}\right\} \tag{4.23}$$

在 SLAM 过程中，机器人位姿的 6 个自由度之间的运动估计精度均为σ且相互独立，则协方差矩阵为对角线上元素为σ^2的 6*6 的对角阵，因为位姿约束信息矩阵是协方差矩阵的逆，所以帧间位姿约束信息矩阵Ω_{ij}如下式(4.24)所示：

$$\Omega_{ij}=\begin{bmatrix}\sigma^{-2} & 0 & 0 & 0 & 0 & 0\\ 0 & \sigma^{-2} & 0 & 0 & 0 & 0\\ 0 & 0 & \sigma^{-2} & 0 & 0 & 0\\ 0 & 0 & 0 & \sigma^{-2} & 0 & 0\\ 0 & 0 & 0 & 0 & \sigma^{-2} & 0\\ 0 & 0 & 0 & 0 & 0 & \sigma^{-2}\end{bmatrix} \tag{4.24}$$

4.3.4 图优化与 g2o

错误闭环检测和误差较大的运动估计所对应的约束边都导致整个图优化模型产生错误的约束力，大大影响其优化精度。有两种方法避免这类情况：第一种方法是通过对约束边所产生的误差进行降维，减小某些误差过大的边对全局优化的影响；另一种方法是比较对应帧之间优化前后位姿运动变换 T_{ij} 的大小，在优化图中去掉前后变化过大对应约束边，再对处理过的优化图模型进行二次优化。

g2o[58]全称是 General Graph Optimization，即是通用图优化，其是由 C++语言编写的专门用于求解非线性最小二乘问题的算法开源库。g2o 的核里有各种各样的求解器，它的顶点和边的类型多种多样，而且 g2o 可以求解 Bundle adjustment，ICP，数据拟合等问题。g2o 库的基本结构，如图 4-2 所示。由图 4-2 可知，SparseOptimizer 既是一个 Optimizable Graph，也是 Hyper Graph。一个 SparseOptimizer 含有很多个顶点和很多个边，可以通过 SparseOptimizer.addVertex 和 SparseOptimizer.addEdge 向一个图中添加顶点和边，然后调用 SparseOptimizer.optimize 完成优化。SparseOptimizer 通过 Optimization Algorithm 指定迭代算法，该 Optimization Algorithm 类可继承 Gauss-Newton，Levernberg-Marquardt 和 Powell's dogleg 三种迭代算法之一。Optimization Algorithm 通过 Solver 申请两个求解器，一个是用于计算稀疏的雅可比和海塞的 SparseBlockMatrix，另一个是用于计算式(4.16)的线性方程求解器 LinearSolver，且该求解器可以是 PCG，CSparse 和 Choldmod 三者之一。由上述可知，在 g2o 中选择优化方法分三个步骤：

1.选择一个线性方程求解器，从 PCG，CSparse 和 Choldmod 三者中选之一。

2.选择一个求解器 BlockSolver。

3.选择一个迭代算法策略，从 GN，LM 和 Doglog 三者中选之一。

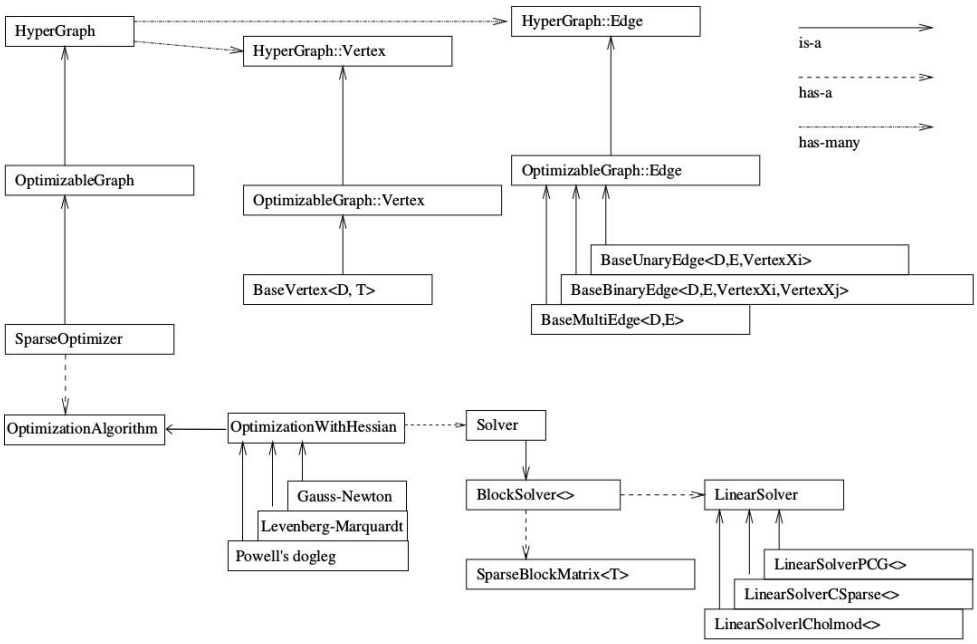


图 4-2 g2o 库的基本结构图

Fig.4-2 The basic structure chart of g2o

4.4 视觉 SLAM 的闭环检测

从前几个章节叙述可知，机器人在运动过程中，在获取图像数据、特征提取与匹配和运动估计等几个方面都会存在误差。即使我们使用 Bundle adjustment 进行局部的或者全局的优化，仍然会存在累积误差。机器人运动的时间越长，这种误差影响越明显。那么，通过闭环检测对所有姿态结果进行优化，这是消除累计误差最有效的方法，闭环检测是一个比 Bundle adjustment 更加强烈、更加精准的约束。

自基于图优化的 SLAM 理论形成以来，有了很多闭环检测的方法被提出，主要分四种：

1.暴力检测的方法，将新获取的关键帧与之前所有的关键帧一一比较，整体运算效率相当低，严重影响了视觉 SLAM 的实时性。

2.基于随机检测的方法，从之前所有关键帧中随机选取 N 帧，选出其中与当前帧相似的帧，但当构建场景地图面积较大，导致闭环检测运算效率很低。

3.基于 KD-Tree 的方法，先是将特征构建成 KD-Tree，这样可以用来与当前帧进行检索对比，此方法是目前主流的检测方法之一。

4.基于 BOVW 的方法，BOVW（视觉词袋）模型最早出现在神经语言程序学(NLP)和信息检索(IR)领域，近年来被广泛应用于计算机视觉中。基于 BOVW 模型的闭环检

测的运算效率高，且适用于大规模场景的地图创建。

本文采用基于 BOVW 模型的闭环检测方法，可描述机器人运动过程获取的场景特征，有效提高了计算和识别视觉外观场景的效率和准确性。在基于 BOVW 模型的闭环检测方法中，每张关键帧为地图上的节点，每个节点用当前对应的场景外观建模，先是提取图像序列中关键帧图像的局部 ORB 特征点，使用 K-means 聚类方法对提取的 ORB 特征点进行分为若干类，每一个类相当于一个视觉单词，这样就将连续的场景特征点离散化为很多的视觉单词。因为每一张图像由很多视觉词汇组成，所以然后我们利用统计的方式将建立的视觉词典生成直方图，最后采用基于概率的方法或者相似性匹配的方法来衡量两幅关键帧图像的相似度。

4.5 本章小结

由于噪声的存在，为了得到长时间的最优运动轨迹和地图，本章叙述了两种视觉 SLAM 后端：基于滤波器和基于图优化，详细介绍了基于卡尔曼滤波和基于粒子滤波的 SLAM 的工作流程，也说明了基于滤波器的视觉 SLAM 很难被用于大型环境的地图建立等缺点。本文将采用图优化作为后端优化方法，先是详细叙述了如何构建 SLAM 图优化模型，然后解释了求解图优化问题的推导过程，接着介绍了如何使用 g2o 开源库来实现图优化，最后介绍了在视觉 SLAM 中闭环检测的方法，而且叙述了基于 BOVW 模型的闭环检测方法的实现流程。

第五章 移动机器人自主探测的实现

虽然基于视觉 SLAM 的移动机器人能够实现在未知环境下即时定位与构建地图，但是其需依赖于人工手动控制移动机器人进行导航过程中构建环境地图。自主探测能够解决自动导航并构建场景环境地图问题，这也是移动机器人很重要的任务[59]。移动机器人位于未知环境时测量其身处的环境，以及自主决定行走到下一位置收集尽可能多的环境信息来构建地图。为了移动机器人能够自主探测未知环境，本文实现了一种基于边界的探测策略。移动机器人自主探测的导航过程中，涉及了全局路径规划和局部路径规划，本章也将对路径规划作详细介绍。

5.1 全局路径规划算法

路径规划是指在有障碍物且无规则的环境场景中，按照一定的评价标准，寻找一条从起始姿态到目标姿态的最优无碰撞路径。路径规划技术分为两大类[60]：一是基于已知周围环境信息的基础上对移动机器人的导航进行路径规划，此规划类型为全局路径规划；二是基于多种传感器信息的基础上对移动机器人在未知环境下的导航进行路径规划，此规划类型为局部路径规划。目前主流的全局路径规划算法主要有 Dijkstra 算法和 $\mathbf{A}^*$ 算法。

5.1.1 基于 Dijkstra 算法的全局路径规划

Dijkstra（迪杰斯特拉）算法是由荷兰计算机科学家狄克斯特拉于 1959 年提出的，是计算从一个节点到其余各节点的最短路径，能在有向图中寻找出最短路径。Dijkstra 算法的主要特点是以起始点为中心向外层层节点扩展搜索，直至搜索到终点为止，其也是广度优先算法的一种。

Dijkstra 算法的输入是一个带权重的有向图 G=(V,E)，V 表示有向图中节点的集合，E 表示节点之间带权边的集合，每条边 E[i]都有其对应的权值 w[i]，Dijkstra 算法可找到由源点 v 到其余各点的最短路径。

基于 Dijkstra 算法的路径规划在执行过程中，把图中节点集合 V 分成两组，第一组为已确定最短路径的节点集合，用 S 表示。初始时 S 中只有一个源点 v，当求得一条最短路径时，便将该节点添加到集合 S 中，直至所有节点都添加到 S 中，则 Dijkstra

算法结束。第二组是其余尚未确定最短路径的节点集合，用 U 表示。按最短路径长度递增的顺序依次将第二组的节点添加到 S 中，在添加过程中，一直保持从源点 v 到 S 中各节点的最短路径长度不大于从源点 v 到 U 中任何节点的最短路径长度。另外，每个节点都有一个距离值，S 中节点的距离等于从 v 到该节点的最短路径长度，U 中节点的距离等于从 v 到此节点只包括 S 中的顶点为中间顶点的当前最短路径长度。

Dijkstra 算法的步骤：

(1)初始化时，节点集合 S 中只有一个源点 v，集合 U 包含除了初始 v 以外的其余节点。若 v 与集合 U 中节点 u 之间有边，则 v 与 u 的距离为边<u,v>的权值。将集合 U 中不是 v 的邻接点的节点距离设置为∞。

(2)从 U 中选择一个与 v 距离最小的节点 k，并将 k 放入到集合 S 中。

(3)以 k 为中间节点，修改集合 U 中与 k 邻接的节点距离。若从 v 到达该节点(不经过 k)比经过 k 的距离长，则将该节点放入到集合 S 中，并且修改集合 U 中已经搜索过的节点路径，修改后的距离值是中间点 k 的距离加上边<k,u>的权值。

(4)重复(2)和(3)的步骤，直至找到目标点，推算出最短路径。

5.1.2 基于 A*算法的全局路径规划

A* 算法是由 P. E. Hart 等人于 1968 年提出的一种启发式搜索算法[61]。启发式搜索是指从当前节点搜索下一个节点时，可以通过设置一个启发函数在全局地图信息中进行选择，以代价最小的节点为下一个搜索节点，如果同时有一个以上代价最少的节点，那么就任选其一。这种算法避免了很多徒劳无功的路径搜索，减小了搜索空间，提高了搜索效率。在启发式搜索中，对位置的评估至关重要，而且采用不同的评估会产生完全不一样的效果。**A*** 算法的估价函数如下式(5.1)所示：

$$f(a) = g(a) + h(a) \tag{5.1}$$

其中，$f(a)$ 为从起始位置经过节点 a 到达目标位置的最低代价的估计值；$g(a)$ 为在状态空间中从起始位置到当前位置节点 a 的实际代价值；$h(n)$ 为从当前位置节点 a 到目标位置最佳路径的估计代价值，即是启发函数。

A* 算法使用欧几里得距离来计算从起始位置到当前位置节点 a 的实际代价值，如式(5.2)所示，使用经典 Manhattan 距离来估计当前位置节点 a 到目标点的代价值，如式(5.3)所示。

$$g(a)=\sqrt{(s.x-a.x)^2+(s.y-a.y)^2} \tag{5.2}$$

$$h(a)=|a.x-g.x|+|a.y-g.y| \tag{5.3}$$

其中，$(s.x,s.y)$、$(a.x,a.y)$和$(g.x,g.y)$分别为移动机器人的起始位置点、当前位置点和目标位置点在全局坐标系中的位置。

基于$\mathbf{A}^*$算法的路径规划的实现步骤：

(1)创建环境模型，初始化栅格地图；

(2)创建搜索列表开启列表 OpenList 和关闭列表 CloseList，初始化搜索列表和估价函数式(5.1)，然后将起始位置节点加入到开启列表 OpenList；

(3)判断 OpenList 列表是否为空，如果 OpenList 列表为空，则跳转到步骤(5)；如果 OpenList 列表不为空，遍历 OpenList 列表查找出估价函数$f(a)$值最小的节点，当有相同值的节点则随机选其一，并将该节点设置为当前节点 cnode，再将该节点加入到 CloseList 列表中，对与当前节点 cnode 相邻的每个节点进行如下操作：

a.如果该节点为不可行区域或已经在 CloseList 列表中，那么无需任何操作；否则，继续步骤 b；

b.如果该节点不在开启列表 OpenList 中，将其添加到 OpenList 列表。把当前节点 cnode 设置为该节点的父节点，并记录该节点的$f(a)$、$g(a)$和$h(a)$的值；

c.如果该节点已在开启列表 OpenList 中，用$g(a)$值作为参考，判断当前节点 cnode 到达该节点的路径是否距离更短，$g(a)$值越小表示路径越短。如果该$g(a)$值更小，则把该节点的父节点改为当前节点 cnode，并重新计算、记录该节点的$f(a)$和$g(a)$值。最后，按$f(a)$值的大小将开启列表 OpenList 内的节点进行重新排序；

(4)将节点 cnode 加入到关闭列表 CloseList，如果目标节点在 CloseList 列表中，则跳转到步骤(6)，否则跳转到步骤(3)；

(5)此次路径规划失败，输出失败信号，跳转到步骤(7)；

(6)从目标节点开始，沿着父节点指向，依次移动到起始位置节点。该路径为搜索到的路径，保存路径并输出；

(7)规划结束。

5.2 避障处理的局部路径规划算法

路径规划与避障是有相关性的两个问题。路径规划主要解决移动机器人整体行走

路径的优劣程度问题，而避障主要解决移动机器人行走时如何处理避开障碍物的问题，即是大多数情况下路径规划都会涉及避障问题，路径规划也是为了使移动机器人避开障碍物。局部路径规划是指在环境信息完全未知或部分已知的情况下，移动机器人通过自身的传感器设备来获取环境信息与自身状态信息，根据得到的传感器信息在线规划实时路径进而躲避障碍物。局部路径规划与全局路径规划对比，其在实时规划方面具有更好的环境适应性和动态避障特性，非常适用于未知且动态的环境。目前，典型的局部路径规划方法有三类：人工势场法、遗传算法、动态窗口法。

5.2.1 基于人工势场法的局部路径规划

人工势场法(Artifical Potential Field)是由 Khatib 博士于 1986 年提出的一种简单易行的路径规划算法[62]。该算法最初是用于机器人的机械臂部位的避障运动规划中，实现机械臂的实时避障。随着国内外学者的深入研究，逐渐将人工势场法适用在机器人导航的路径规划中。

在移动机器人工作规划的模型空间中，目标位置为q_g，那么目标对移动机器人作用的引力势为$U_{att}(q)$；障碍物位置为q_{obs}，那么障碍物对移动机器人作用的斥力势为$U_{rep}(q)$；移动机器人位于位置q的势场强度为U_q，且其势场为引力势和斥力势的总和，如下式(5.4)所示：

$$U(q)=U_{att}(q)+U_{rep}(q) \tag{5.4}$$

当$U(q)$极小时，处于位置q的移动机器人所受合力为合势场的负梯度，合力为式(5.5)所示。

$$\vec{F}=-\nabla U(q)=-\nabla U_{att}(q)-\nabla U_{rep}(q) \tag{5.5}$$

其中，$-\nabla U(q)$为U在位置q的梯度，其方向为工作空间中位置q处势场变化最大的方向。二维环境空间的任意位置$q(x,y)$的梯度如式(5.6)所示：

$$\nabla U(q)=\begin{bmatrix}\dfrac{\partial U}{\partial x}\\ \dfrac{\partial U}{\partial y}\end{bmatrix} \tag{5.6}$$

本文采用静电场势场模型来定义势场函数$U(q)$，那么工作空间中引力场函数为：

$$U_{att}(q)=\frac{1}{2}\xi\rho^2(q,q_g) \tag{5.7}$$

其中，ξ为引力场的比例系数，$\rho^2(q,q_g)$为移动机器人当前位置到目标位置距离的平方。

另外，斥力场如下式(5.8)所示：

$$U_{rep}(q)=\begin{cases}\frac{1}{2}\eta\left(\frac{1}{\rho(q,q_{obs})}-\frac{1}{\rho_0}\right)^2, \rho(q,q_{obs})\le\rho_0\\ 0, \rho(q,q_{obs})>\rho_0\end{cases} \tag{5.8}$$

其中，η为斥力场的比例系数，$\rho(q,q_{obs})$为移动机器人当前位置到障碍物的距离，ρ_0为障碍物对移动机器人影响的最小距离，只有当移动机器人与障碍物的距离小于ρ_0时，障碍物才对移动机器人产生斥力。

综上所述，移动机器人所受到的引力与斥力分别如式(5.9)和式(5.10)所示，可知引力和斥力两者的合力决定移动机器人在规划工作空间中的走向。

$$\vec{F}_{att}=-\xi\rho(q,q_g) \tag{5.9}$$

$$\vec{F}_{rep}=-\nabla U_{rep}(q)=\begin{cases}\eta\left(\frac{1}{\rho(q,q_{obs})}-\frac{1}{\rho_0}\right)\frac{\nabla\rho(q,q_{obs})}{\rho^2(q,q_{obs})}, \rho(q,q_{obs})\le\rho_0\\ 0,\ \ \rho(q,q_{obs})>\rho_0\end{cases} \tag{5.10}$$

5.2.2 基于遗传算法的局部路径规划

遗传算法（Genetic Algorithm，简称 GA）是由 John Holland 教授等人[63]于 1988 年提出的一种搜索算法，Holland 等人了解分析出生物的遗传和自然进化与人工自适应系统有相似关系，研究模拟了生物个体对自然环境的适应进化过程，构建了模拟生物遗传进化机制的人工模型，再以群体的形式来自适应搜索，形成了较完善的遗传算法理论体系。遗传算法的主要特点：(1)具有并行性和并行计算的搜索能力。(2)具有良好的可扩展性，鲁棒性强。(3)具有自组织性和自适应性。(4)遗传算法的运算对象是问题解本身的某种编码形式。(5)采用概率转换方式。(6)不需要辅助信息或求导。(7)对于确定的问题，遗传算法可求解出很多潜在解集，使用者可根据需求选择解。(8)遗传算法具有处理多目标问题的能力。

在局部路径规划问题中，由于对路径好坏的判断有很多标准，可采用遗传算法对所有要求标准进行设定，来引导种群的最优进化方向，最终获得满足所有标准的结果，即是最佳避障路径。基于遗传算法的局部路径规划的基本流程分为六个步骤：

1.个体编码。遗传算法求解问题首先要明确编码和解码的方式，即是将染色体与目

标问题间构建一种映射关系。在特定的评判原则下，有些染色体二进制字符串会被归类为不良个体，但其仍具有潜在优良基因的可能性。在经过一系列的遗传操作后，这些潜在的优良基因可能转变成优良个体，遗传给后代，最终获得最优解。

2.初始种群的生成。根据目标问题的不同情况，我们需要实际地设定在几十到几百个体数量之间的种群规模。采用随机生成种群的方式，搜索潜在可行解，这样能够适当降低陷入局部最优和早熟的可能性。该方式适用于任何目标问题，覆盖范围广，但可能减缓收敛速度。因此，我们也可以通过采取案例搜索法、贪婪算法等启发式算法生成初始种群，这样更好地提高收敛速度。

3.适应度函数的设定。遗传算法通过设定恰当的适应度函数，来矫正和控制种群朝某一特定优良方向进化，同时限制了不良个体的繁殖。适应度函数是通过目标函数来确定的，目标函数的值可能是正值或者是负值，其值为正值时表示适应度函数求的是最大值，其值为负值时表示适应度函数求的是最小值。

4.遗传算子。遗传算法包括了选择算子、交叉算子和变异算子这三个基本遗传操作，这三者在遗传算法中所起的作用是各有不同。选择算子是根据个体的适应度值的大小来判断该个体是否可保留下来。交叉算子是对从群体中选出的两个的染色体进行采取部分基因信息互换的操作，进而产生两个新的个体，增加了优秀解出现的可能性。变异算子是改变个体染色体的个别基因，目的是把我们本来不想要的这些基因向好的方向转化。

5.控制参数选择。控制参数选择在遗传算法中是至关重要的，选择不同的控制参数对遗传算法性能的影响是非常大的，乃至整个遗传算法的收敛性都会受到严重影响。控制参数主要包括二进制(十进制)编码长度、群体规模 N 、变异概率 p_m 、交叉概率 p_c 等。

6.算法终止条件。任何一种进化算法都不能无限地运行下去，必须在合理的时间范围内终止，因而引进了终止机制。遗传算法通常用到的终止标准：

收敛标准：当遗传算法执行到一定程度的时候，存在于群体中的个体字符串的构成很相似，难以有比这些更好的个体出现。

精度标准：当个体的评价函数的数值满足规定的要求时说明达到了标准，则停止算法搜索。

时间标准：当运算时间或者迭代次数超过规定阀值时，则遗传算法就终止运行。

5.2.3 基于动态窗口法的局部路径规划

动态窗法[64]是将有限的速度和加速度的运动约束限定在一个可行的动态范围内，在速度(v,w)空间中随机选取多组速度，接着模拟移动机器人在这多组速度下在一段时间内的运动轨迹。对获得的多组运动轨迹进行评价，进而选取最优轨迹所对应的速度来驱使移动机器人行走。基于动态窗口法的局部路径规划涉及三个关键步骤：

1.机器人的运动模型

在动态窗口法中，首先需要已知机器人的运动模型，才能模拟出机器人的轨迹。该运动模型是假设双轮机器人的轨迹是一段段的圆弧或者直线，某一时刻速度(v_t,w_t)表示一个圆弧轨迹。

假设机器人只能前进和旋转，由于移动机器人在两个相邻时刻内的运动距离很短，我们可以将两个相邻点之间的运动轨迹认为是直线，即在机器人坐标系下移动机器人的运动距离x轴为$v_t*\Delta t$，如图 5-1。在世界坐标系下，$t+1$时刻相对于t时刻机器人移动的位移Δx和Δy：

$$\begin{aligned}\Delta x &= v\Delta t\cos(\theta_t)\\ \Delta y &= v\Delta t\sin(\theta_t)\end{aligned} \tag{5.11}$$

将一段时间内的位移增量累计求和：

$$\begin{aligned}x &= x+v\Delta t\cos(\theta_t)\\ y &= y+v\Delta t\sin(\theta_t)\\ \theta_t &= \theta_t+w\Delta t\end{aligned} \tag{5.12}$$

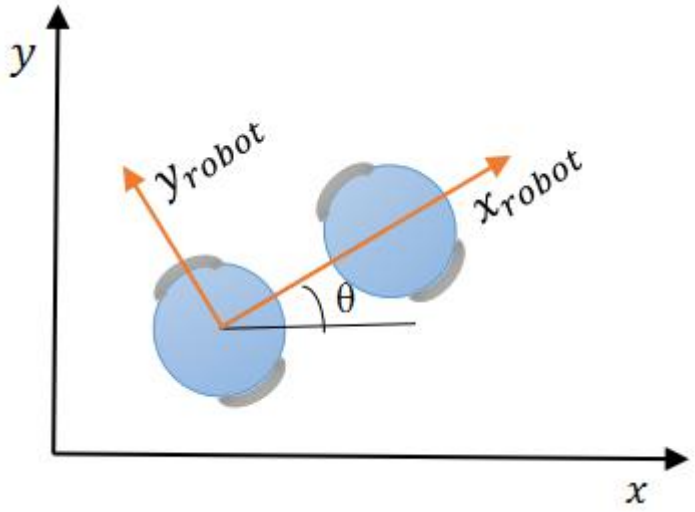


图 5-1 计算运动距离

Fig.5-1 Calculating the distance of movement

2.速度采样

在已知的机器人的运动模型下，我们可以根据速度推算出运动轨迹。由于在速度(v,w)的二维空间中会存在无数多组速度，我们需要根据移动机器人自身以及环境的限制将采样速度控制在一定范围之内。当考虑到移动机器人受到自身最大速度和最小速度的限制，采样速度如式(5.13)所示。

$$V_m = \{v \in [v_{\min}, v_{\max}], w \in [w_{\min}, w_{\max}]\} \tag{5.13}$$

由于电机力矩有限，会存在最大的加减速限制，那么移动机器人轨迹前向模拟的动态窗口内的实际速度为采样速度，如式(5.14)所示。

$$V_d = \left\{(v,w) \mid v \in \left[v_c - \dot{v}_b \Delta t, v_c + \dot{v}_a \Delta t\right] \wedge w \in \left[w_c - \dot{w}_b \Delta t, w_c + \dot{w}_a \Delta t\right]\right\} \tag{5.14}$$

其中，v_c和w_c为移动机器人的当前速度，v_b和w_b为移动机器人的最大减速度，v_a和w_a为移动机器人的最大加速度。

为了使移动机器人在障碍物前停下避障，在最大减速度情况下，采样速度也有一定范围：

$$V_a = \left\{(v,w) \mid v \le \sqrt{2 \cdot dist(v,w) \cdot \dot{v}_b} \wedge w \le \sqrt{2 \cdot dist(v,w) \cdot \dot{w}_b}\right\} \tag{5.15}$$

其中，$dist(v,w)$为速度(v,w)对应轨迹上离障碍物最近的距离。

3.评价函数

由于在多组采样速度下的多组运动轨迹都是可行，我们需要采用评价函数来为每条轨迹进行评价，选取最优运动轨迹。评价函数如下式(5.16)所示：

$$G(v,w) = \sigma(\alpha \cdot heading(v,w) + \beta \cdot dist(v,w) + \gamma \cdot velocity(v,w)) \tag{5.16}$$

其中，$heading(v,w)$为方位角评价函数，用来评价移动机器人以当前的采样速度到达模拟轨迹末端时的朝向和目标之间的角度差距。$dist(v,w)$表示移动机器人在当前轨迹上与最近障碍物的距离。$velocity(v,w)$表示当前轨迹的速度大小。

5.3 基于边界的探测策略

为了实现移动机器人自主探测未知环境并建立地图，本文提出一种基于边界的探测策略，边界是指已知区域与未知区域之间的边缘处。我们先是在现有的栅格地图上检测出边界栅格，然后驱使移动机器人运动到距离最近的边界上，进而探测未知领域

并扩展全局栅格地图的建立，直至没有边界栅格，即是探测完未知环境。

如果移动机器人依赖一个完善的地图能够导航至空间中一个特定的点，那么该点被认为是可到达的。因为每一条路径都是从移动机器人的起始位置到每个可到达的点，所以可知整个可到达的空间是连续的。从移动机器人的起始位置开始构建环境地图，每条路径都至少是建图过程中的一部分。其主要思路是根据当前所获得的环境地图来选取移动机器人将要到达的下一系列探测点，接着使用估价函数对一系列候选探测点进行评估，并从中选取估价值最大的候选点作为下一探测点。当移动机器人导航至边界探测点过程中，它将沿着路径获取更多环境信息并构建地图。

5.3.1 边界检测

构建好当前探测到环境的栅格地图之后，通过将栅格地图上每个单元的占用概率与初始概率进行比较来分类。对于每个单元，如果其占用概率大于 0.6，则称为被占单元，即是视觉传感器已经探测且存在障碍物的领域；如果其占用概率小于 0.4，则称为自由单元，即是视觉传感器已经探测且没有障碍物的领域；如果其占用概率大于 0.4 且小于 0.6，则称为未知单元，即是视觉传感器尚未探测的领域。任何与未知单元相邻的自由单元都标记为边界单元，围绕边界单元的相邻单元都被划分为边界域。

本文在边界检测获取方面，先是在当前的栅格地图上检测获取一个边界单元，标记该边界单元属于边界域的部分，并检测该单元周围 8 个相邻单元是否为边界单元。如果其相邻单元为边界单元，则该相邻单元标记归纳为边界域，并继续检测该相邻单元周围 8 个相邻单元是否为边界单元，直至没有相邻单元为边界单元，则确定了一个完整的边界域。之后，重新在当前的栅格地图上检测获取新的边界单元，重复上述步骤，以此循环下去，一直到标记出全部边界单元为止。所有边界域都被检测提取出了，边界域检测的伪代码如算法 5.1 所示。我们需要计算该算法检测到的每个边界域的大小，如果一个边界域的大小大于规定的阀值，则计算该边界域的中心点 $c(x,y)$，否则该边界域被丢弃。计算边界域的中心点如式(5.17)。

$$\begin{aligned} c_x &= \frac{1}{N_{region}} \cdot \sum_{i=1}^{N_{region}} x_{i,region} \\ c_y &= \frac{1}{N_{region}} \cdot \sum_{i=1}^{N_{region}} y_{i,region} \end{aligned} \tag{5.17}$$

其中，N_{region} 为边界域中边界单元数目。

算法 5-1 在 2D 栅格地图上进行边界域检测
Algorithm 5-1 Extraction of frontier regions on 2d occupancy grid

1: **procedure** 2DGridFrontierRegionExtraction(f_{cells})
2: $inspect = \phi$ //该队列用于保存已经检测到的边界单元
3: $labeled = \phi$ //此队列表示已经标记归纳为一个边界域中
4: $frontier_{regions} = \phi$ //该队列用于保存每个边界域的一系列边界单元
5: **while** f_{cells} 不为空 **do**
6: $region = \phi$ //该队列用于保存边界域的所有边界单元
7: $inspect = inspect +<$ f_{cells} 队列中最前面的元素>
8: **while** *inspect* 不为空 **do**
9: $cell =$ *inspect* 队列中最前面的元素
10: $inspect = inspect - \langle cell \rangle$
11: $region = region + \langle cell \rangle$
12: $labled = labled + \langle cell \rangle$
13: **for** *cell* 所有相邻的单元 *ngh* **do**
14: **if** *ngh* 为边界单元并且尚未被标记 **then**
15: $inspect = inspect + \langle ngh \rangle$
16: **end if**
17: **end for**
18: **end while**
19: $frontier_{regions} = frontiers_{regions} + \langle region \rangle$
20: **end while**
21: **return** $frontier_{regions}$
22: **end procedure**

图 5-2(a)表示移动机器人探测走廊时所构建的栅格地图；图 5-2(b)表示栅格地图上边界单元部分；图 5-2(c)表示边界域是大于最小的边界单元，每个边界域的中心都用十字线标记。

(a) 走廊的栅格地图　(b) 边界部分　(c) 边界域

(a)The grid map of the corridor　(b) Boundary　(c)Boundary region

图 5-2 栅格地图与边界

Fig.5-2 Grid map and boundary

5.3.2 导航策略

一旦在当前栅格地图上检测完所有边界单元，移动机器人将会尝试导航到最近可到达且未访问的边界单元，查找最近边界单元的伪代码如算法 5.2 所示。利用基于 $\mathbf{A}^*$ 算法的全局路径规划在栅格地图上搜索出从机器人的起始位置到目标边界单元之间无障碍物的最短路径。当移动机器人朝向其目标边界单元移动时，因考虑到动态环境的情况，本文利用基于动态窗口法的局部路径规划进行实时避障，只要室内环境变化不会过大，移动机器人依然会贴近最短路径导航到目标边界单元。

当移动机器人到达目标边界单元，该边界单元被加入到已经访问的边界单元列表中。移动机器人会自主原地旋转 360° 一周，利用 RGB-D 相机捕获新的室内场景信息，进而可以往 3D 点云添加新的图像点云信息，以及更新 2D 栅格地图并为其添加新的导航信息。然后移动机器人在已更新的栅格地图上检测边界单元，再尝试导航至最近可到达且未访问的边界单元。

当移动机器人无法到达目标边界单元，并等待一段时间依然无法到达时，移动机器人会认为该目标点是无法到达的，将该目标边界单元加入到无法到达边界单元的列表中。移动机器人尝试更新栅格地图，重新检测获取一遍边界单元，并导航至最近可到达且未访问的边界单元。

算法 5-2 检测边界
Algorithm 5-2 Detecting frontier

1: **procedure** FrontierBasedExploration(m_t, x_t) // m_t 为当前栅格地图，$x_t = (x, y, \theta)$ 为当前位置

2: $f = \phi$ //该队列用于保存边界单元

3: **for** 所有栅格单元 *cells* **do**

4: **if** 该栅格单元为边界单元 *cell* **do**

5: $f = f + \langle cell \rangle$

6: **end if**

7: **end for**

8: r =2DGridFrontierRegionExtraction(f)

9: **for** 每个边界域$\in r$ **do**

10: **if** $size(region) < size_{\min}$ **then**

11: $r = r - \langle region \rangle$

12: **else**

13: $$c_{region_x} = \frac{1}{N_{region}} \cdot \sum_{i=1}^{N_{region}} x_{i,region}$$

14: $$c_{region_y} = \frac{1}{N_{region}} \cdot \sum_{i=1}^{N_{region}} y_{i,region}$$

15: **end if**

16: **end for**

17: **return** $\min_{x_r, y_r} \left\{ \sqrt{(x - x_r)^2 + (y - y_r)^2} \mid \langle x_r, y_r \rangle \text{为边界域的中心} \right\}$

18: **end procedure**

5.4 本章小结

本章首先阐述了实现移动机器人自主探测的需求和大致过程，然后分别详细介绍了基于 Dijkstra 算法的全局路径规划和基于 **A*** 算法的全局路径规划的实现步骤。用于实时避障的局部路径规划方面，分别详细介绍了人工势场法、遗传算法、动态窗口法。最后，详细描述了基于边界的自主探测策略，阐述了边界检测以及导航策略的实现流程，移动机器人自主探测过程中采用基于 **A*** 算法的全局路径规划搜索最短路径，利用基于动态窗口法的局部路径规划进行实时避障。

第六章 实验与分析

6.1 RGB-D SLAM 算法的实验分析

利用标准测试数据集对 RGB-D SLAM 算法中的特征点提取与描述子计算、特征匹配和运动估计等阶段进行验证对比。同时，也会采用实际室内场景进行测试验证。

6.1.1 实验环境

测试的硬件环境主要包括英特尔第三代酷睿 I5-3570K@3.4GHz 四核处理器，12G 内存，Nvidia GeForce GT640 显卡，华硕的 Xtion PRO 相机。

测试的系统环境为 Ubuntu14.04，内核版本为 Linux 3.16.0-43-generic。

6.1.2 标准测试数据集

为了测试验证的方便性和准确性，本次采用慕尼黑工业大学(TUM)计算机视觉小组提供的 RGB-D 数据集[65]和标准测试程序进行测试 RGB-D SLAM 算法。RGB-D 数据集包括了由 RGB-D 相机获取的 RGB 图像与深度图像，以及由 8 个高速追踪相机所构成的高精度运动捕捉系统所记录 RGB-D 相机的真实位姿信息，即是 ground truth 数据。由于 TUM 的 RGBD 数据集包含了很多系列，本次实验采用的是 rgbd_dataset_freiburg1_room 数据集，该数据集包括了 1360 帧 RGB 和深度图像，以及相应的 ground truth 数据。

在测试本文 RGB-D SLAM 算法时，估计运动轨迹中相机姿态序列表示为 $P_1,\cdots, P_n \in SE(3)$，RGB-D 数据集的真实轨迹中相机姿态序列表示为 $Q_1,\cdots,Q_n \in SE(3)$，这两种姿态的时间和序列长度是相同的。而对 RGB-D SLAM 算法的测试评估内容分为绝对轨迹误差 *ATE* 和相对姿态误差 *RPE* 两部分。

绝对轨迹误差 *ATE* 表示所求得的估计轨迹与真实轨迹之间的偏差。根据计算得到的真实轨迹 Q_i 与估计轨迹 P_i 的刚体变换 T，i 时刻的绝对轨迹误差如式(6.1)所示，全部时刻的均方误差如式(6.2)所示。

$$ATE_i := Q_i^{-1} S P_i \tag{6.1}$$

$$RMSE(ATE_{1:n}) := \left(\frac{1}{n} \sum_{i=1}^{n} \| trans(ATE_i) \|^2 \right)^{1/2} \tag{6.2}$$

相对姿态误差 RPE 表示在固定时间间隔为 Δi 时，运动轨迹的局部正确率，反映了轨迹的变化值，时刻 i 的相对姿态误差如下式(6.3)所示：

$$RPE_i := (Q_i^{-1} Q_{i+\Delta i})^{-1} (P_i^{-1} P_{i+\Delta i}) \tag{6.3}$$

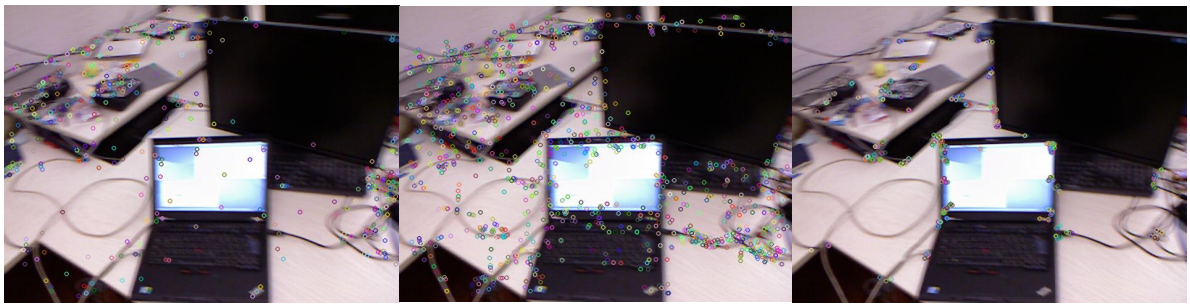
当数据集中图像序列有 n 个相机姿态时，会有 $m = n - \Delta$ 个相对姿态误差，即可得到均方误差：

$$RMSE(RPE_{1:n}, \Delta) := (\frac{1}{m} \sum_{i=1}^{m} \| trans(RPE_i) \|^2)^{1/2} \tag{6.4}$$

其中，$trans(RPE_i)$ 表示相对姿态误差 RPE_i 的平移分量。

6.1.3 特征点提取与匹配的实验分析

先是分别使用 SIFT、SURF 和 ORB 算法对 rgbd_dataset_freiburg1_room 测试数据集第 100 帧 RGB 图像进行特征点检测与描述子计算的测试，测试的结果如图 6-1 所示。对这三种算法的特征点检测时间、描述子计算时间和提取到的特征点数目进行统计与对比，如表 6.1 所示。



(a) SIFT 算法　(b) SURF 算法　(c) ORB 算法

(a) SIFT algorithm　(b) SURF algorithm　(c) ORB algorithm

图 6-1 特征点检测的测试结果

Fig.6-1 Test results of feature points detection

表 6-1 三种算法的测试结果对比

Table 6-1 The test results of the three kinds of the algorithm

算法	特征点检测时间(ms)	描述子计算时间(ms)	特征点数目
SIFT	114.201	133.941	443
SURF	51.236	101.329	1148
ORB	8.158	12.605	500

利用这三种算法进行了特征点检测与描述子计算之后，分别与 RANSAC 相结合对标准测试数据集第 100 帧和第 101 帧 RGB 图像进行特征点匹配，测试结果如图 6-2、图 6-3 和图 6-4 所示。

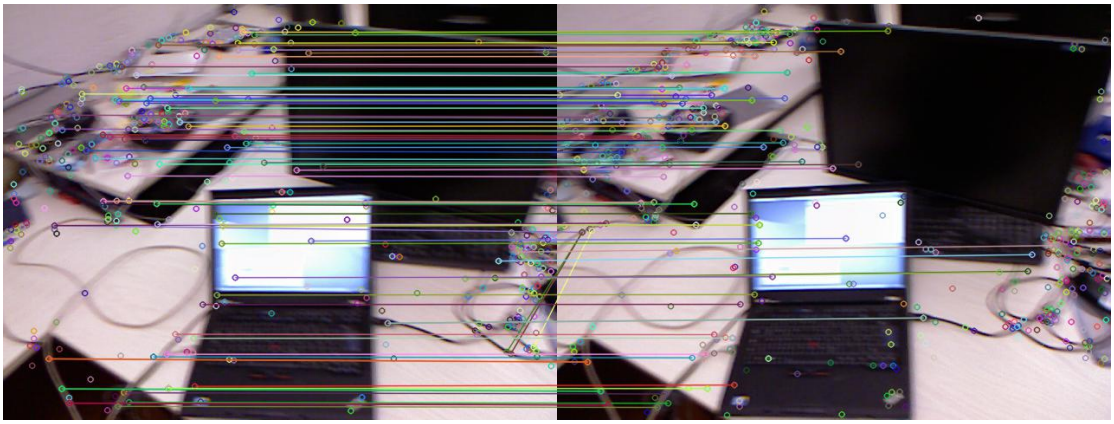


图 6-2 SIFT 特征点提取与匹配

Fig.6-2 Extraction and matching of SIFT feature point

图 6-3 SURF 特征点提取与匹配

Fig.6-3 Extraction and matching of SURF feature point

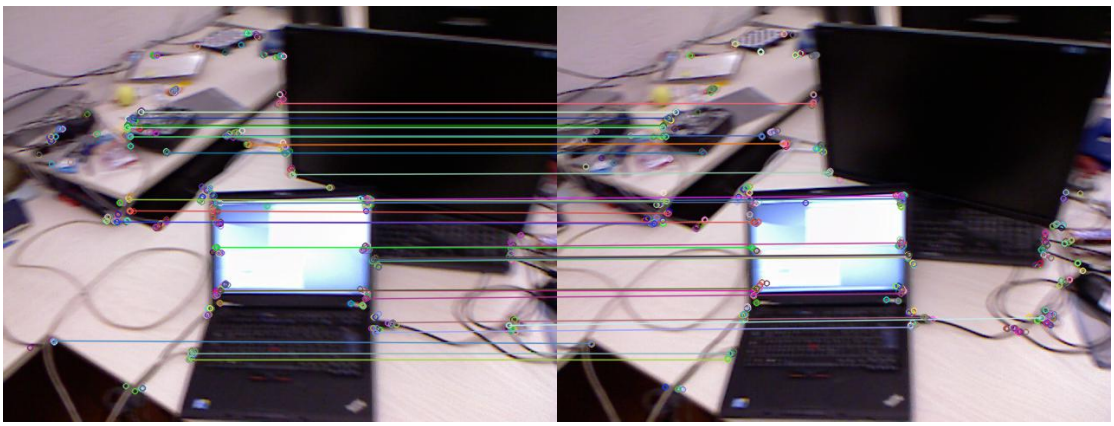


图 6-4 ORB 特征点提取与匹配

Fig.6-4 Extraction and matching of ORB feature point

从上述测试结果可知，在特征点检测与描述子计算方面，ORB 算法比 SIFT 算法、SURF 算法在速度上都要快很多，提取特征点数目适中。在特征点匹配方面，即使与 RANSAC 相结合进行特征点匹配，SIFT 算法和 SURF 算法还是会出现个别的特征点匹配错误，而 ORB 算法并没出现特征点匹配错误。由此可见，ORB 算法在特征点提取与匹配上更具有优势，可以极大地提高基于 RGB-D 的视觉 SLAM 的实现效率。

6.1.4 数据集测试与实际场景测试

RGB-D SLAM 算法利用 rgbd_dataset_freiburg1_room 数据集中每帧图像求解每个时刻相机的旋转矩阵和平移向量，进而可以得到相机的运动估计轨迹。利用式(6.2)和式(6.3)分别计算出估计轨迹与真实轨迹之间的绝对轨迹误差 *ATE* 为 0.052m 和相对姿态误差 *RPE* 为 0.035m，对比效果如图 6-5 和图 6-6 所示。由此可见，运动估计轨迹与真实轨迹没有完全重叠一致，在某些时间点上的相对姿态误差较大，但估计轨迹与真实轨迹的基本形状一致，整体产生的误差是较小且可容忍的，说明了运动估计轨迹效果良好。

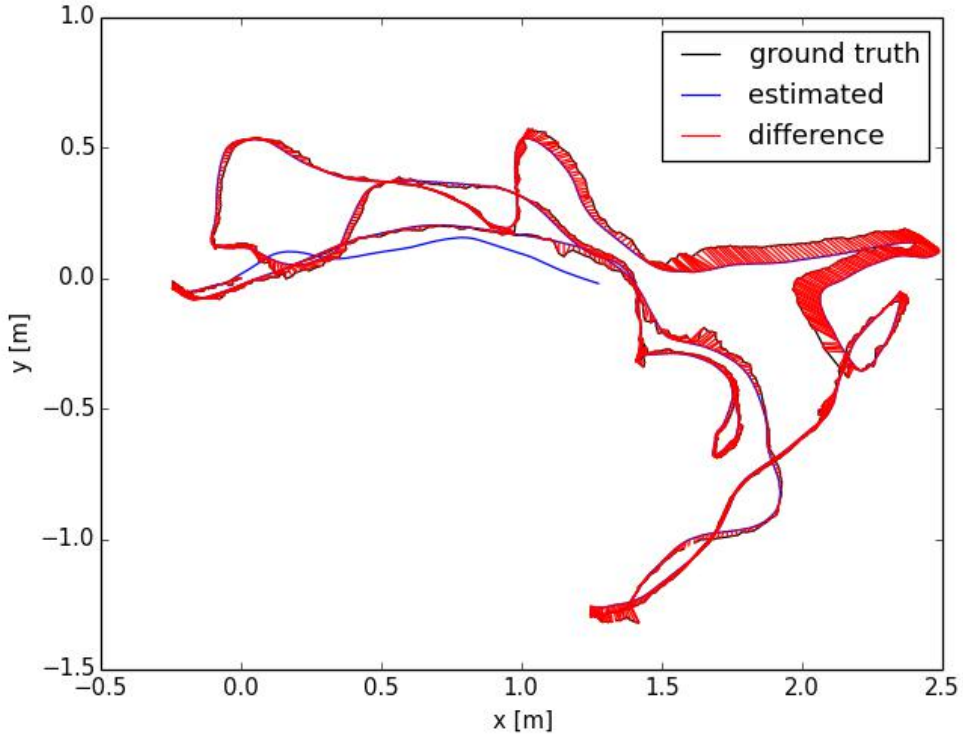


图 6-5 绝对轨迹误差

Fig.6-5 Absolute Trajectory Error

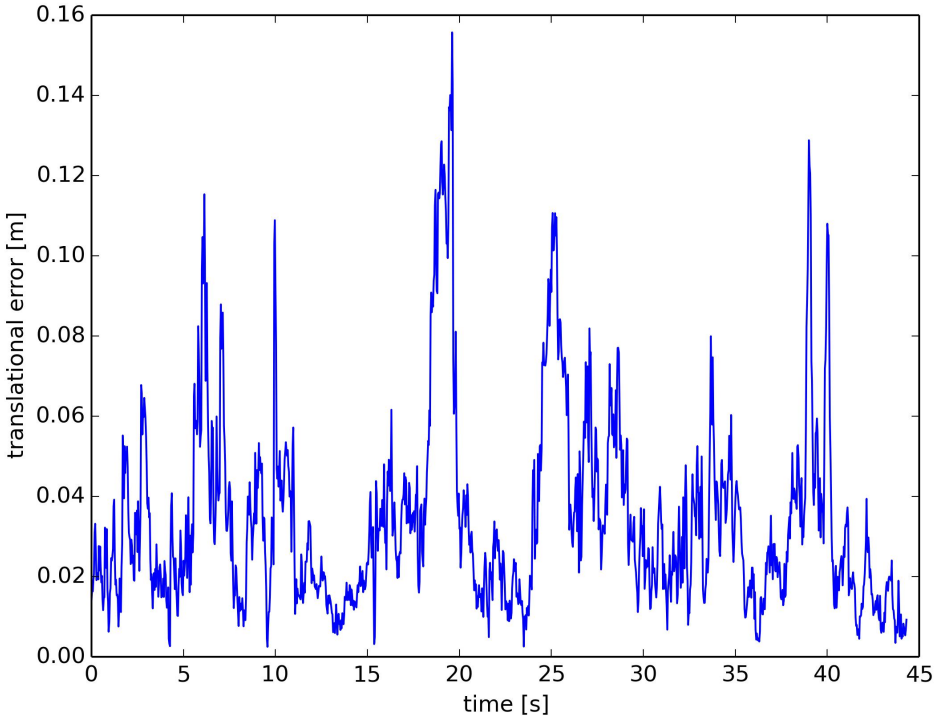
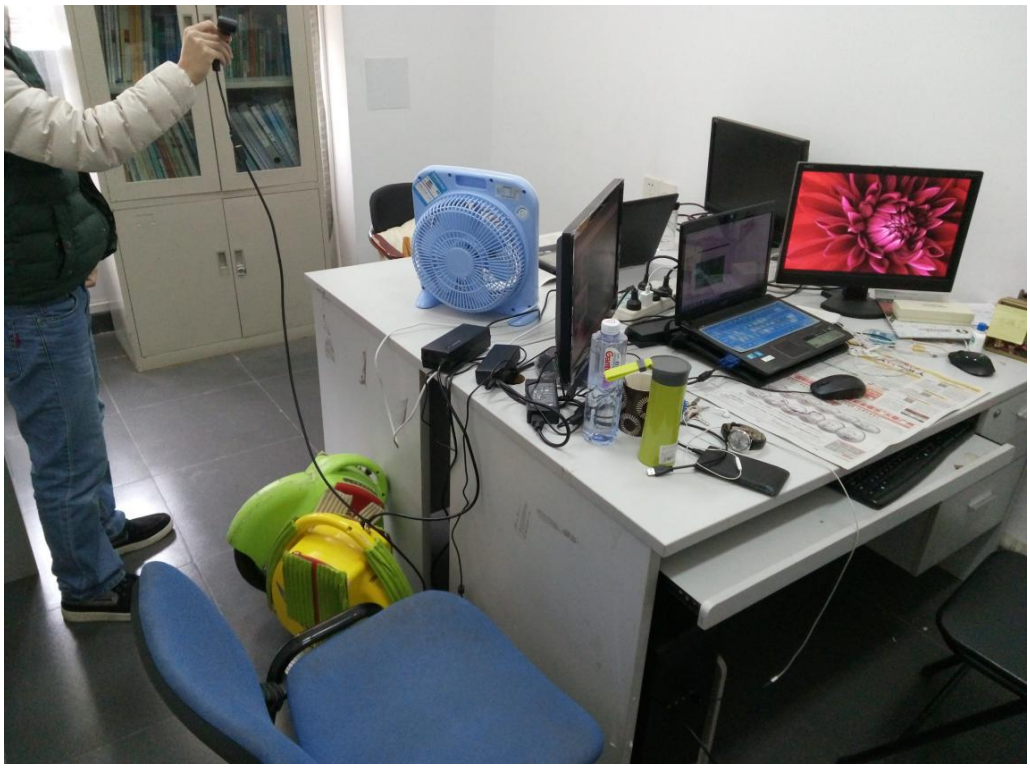


图 6-6 相对姿态误差

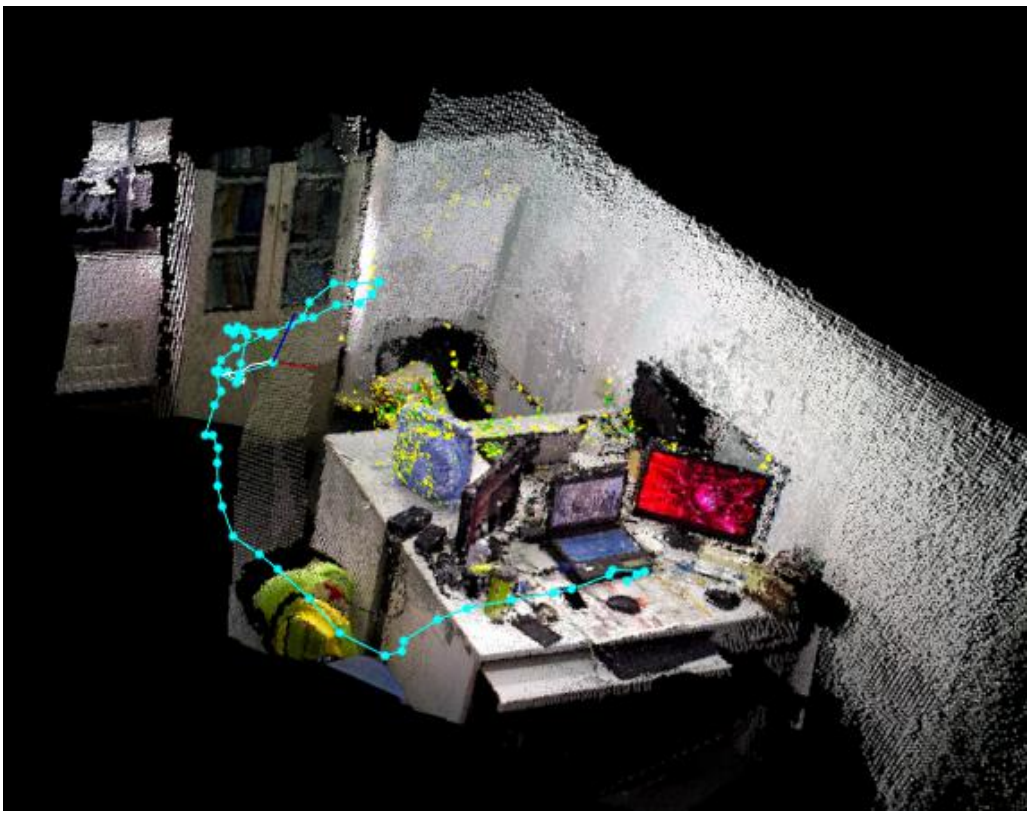
Fig.6-6 Relative Pose Error

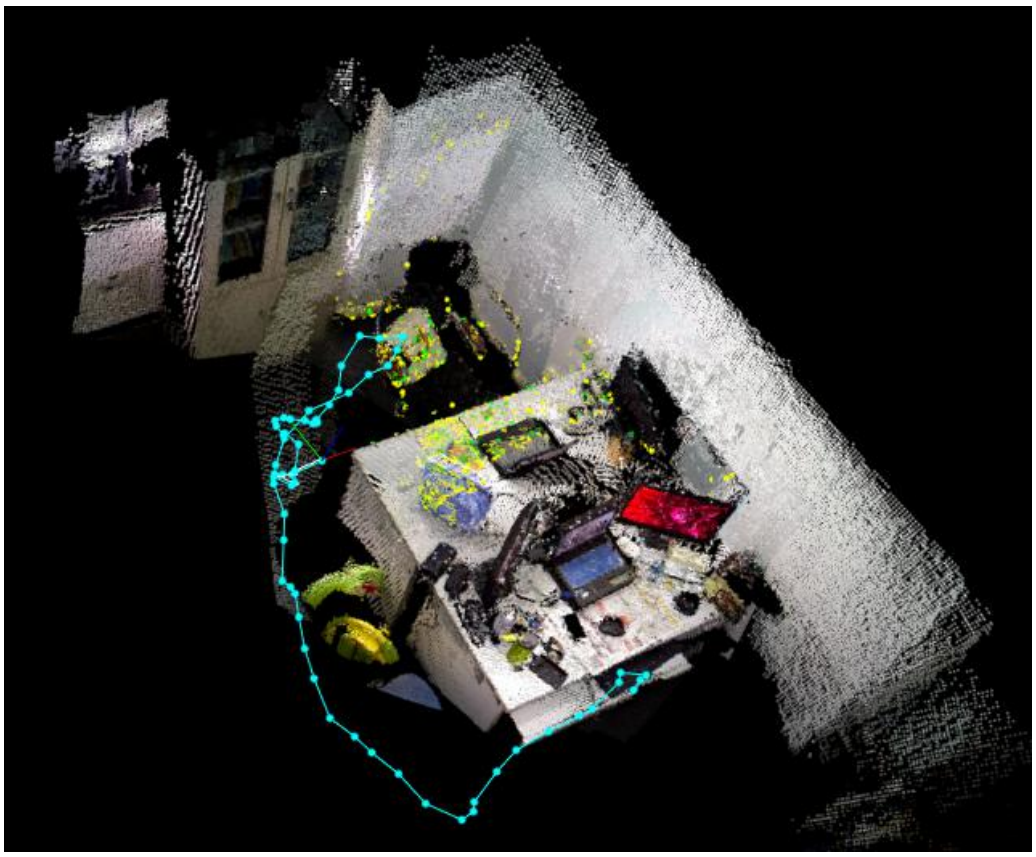
本人还利用华硕 Xtion PRO 相机在办公室实际场景中对 RGB-D SLAM 算法进行验证测试，建立了一个三维点云地图，如图 6-7 所示。



(a)实际场景原图

(a)The original image of the actual scene





(b)实际场景三维点云地图

(b)The 3D point cloud maps of the actual scene

图 6-7 实际场景地图

Fig.6-7 The maps of the actual scene

6.2 移动机器人自主探测室内环境

6.2.1 实验平台配置

在软件方面，采用 Ubuntu14.04 系统中安装 ROS indigo 版本机器人操作系统平台作为软件实验平台，在 Qt Creator 集成环境下利用了便于实现图优化算法的 g2o 库以及用于图像处理的 OpenCV 库等进行软件开发。

在硬件方面，主要有一个经改装过的 TurtleBot2 移动机器人和一台作为上位机的电脑。如图 6-8 所示，TurtleBot2 移动机器人的硬件主要有 Kobuki 移动底盘、华硕 Xtion pro 相机、NVIDIA Jetson TK1 开发板、3000mAh 电池。NVIDIA Jetson TK1 开发板的片上系统(SOC)搭载了包含 192 个 CUDA 核心的 NVIDIA Kepler GPU 和四核 Cortex-A15 架构的 ARM CPU，还有 2GB 运行内存、16GB eMMC 存储空间和 USB 3.0 等配置资源。

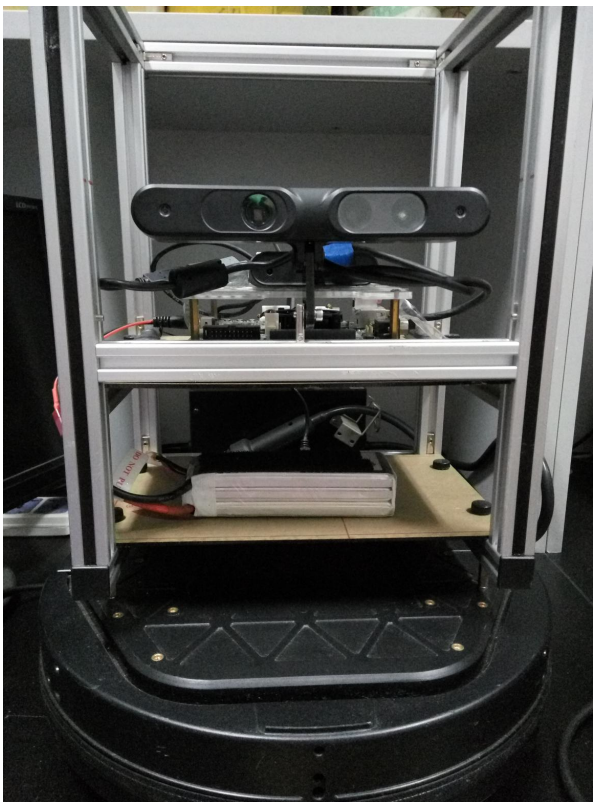


图 6-8 TurtleBot2 移动机器人

Fig.6-8 TurtleBot2 mobile robot

6.2.2 RGB-D SLAM 中视觉里程计的测试

视觉里程计主要是指视觉 SLAM 前端的实现，实验中移动机器人以 0-0.3m/s 的速度环绕约 10*9 米的矩形行驶，将 Kobuki 移动底盘的里程计获取到的二维坐标和行驶距离视为标准值，视觉里程计计算得到的运动轨迹与底盘里程计的运动轨迹作比较，如图 6-9 所示。底盘里程计计算移动机器人行走约 37.54 米，视觉里程计计算得到约 37.88 米，两者误差约 0.9%。可见，两者运动轨迹基本一致，误差不大。

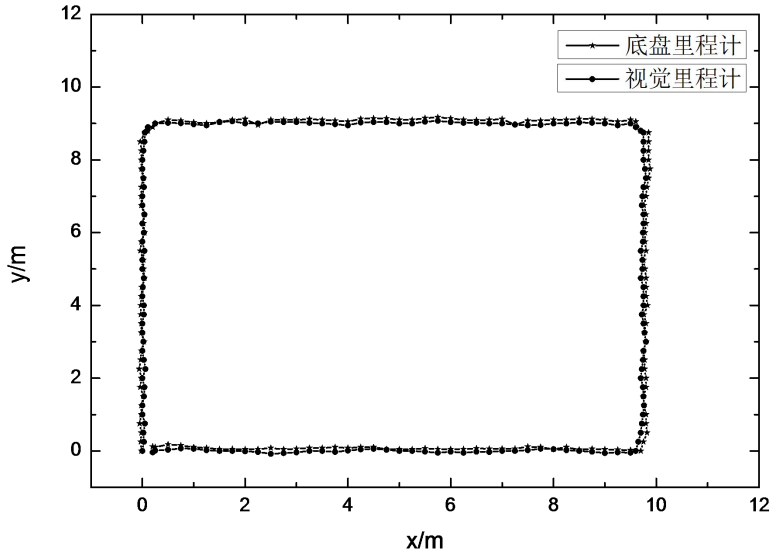
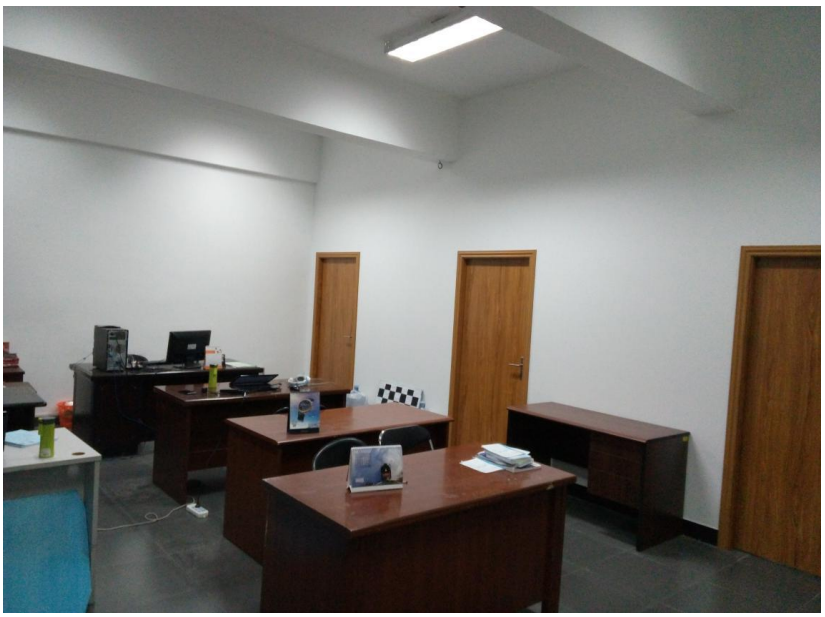


图 6-9 运动轨迹对比图

Fig.6-9 Comparison chart of motion trajectory

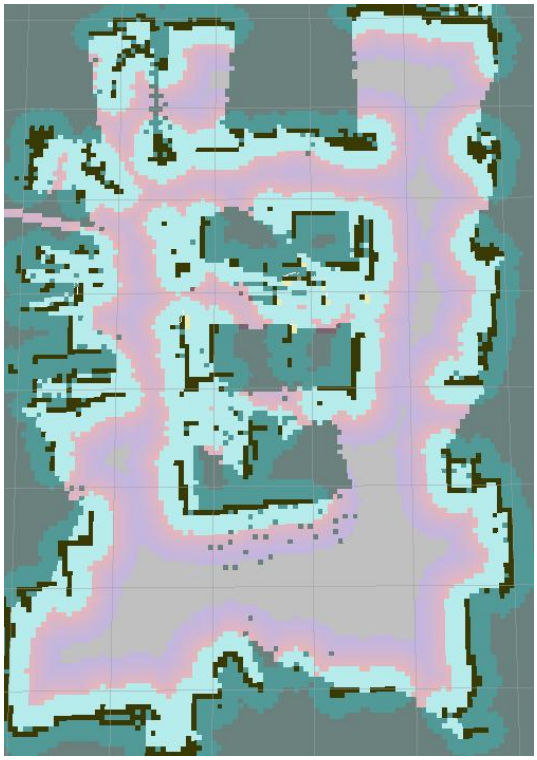
6.2.3 移动机器人探测室内实际场景实验

本次实验环境地点为广工大实验三号楼 304 实验室，TurtleBot2 移动机器人的速度在 0.3 m/s 和 0.4 m/s 之间，旋转速度在 2 rad/s 之内，华硕 Xtion pro 相机以 30hz 的频率采集分辨率为 640*480 的 RGB 图像和深度图像。TurtleBot2 移动机器人在实验室内自主导航，探测未知的场景环境，建立用于导航 2D 栅格地图和 3D 点云地图，如图 6-10 所示。由图可知，图中物体与真实环境中物体基本对齐，整体的栅格地图和点云地图的一致性较好，说明了本文设计的视觉 RGB-D SLAM 算法能较好地估计移动机器人的运动轨迹以及构建与室内场景基本一致的 2D 栅格地图和 3D 点云地图。



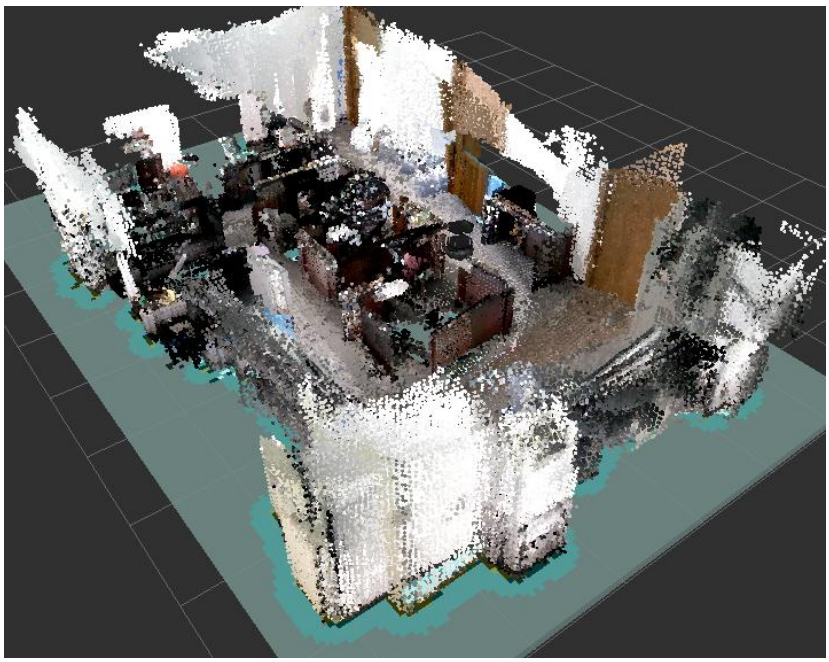
(a)实验室场景原图

(a)The original image of laboratory scene



(b)2D 栅格地图

(b)2D grid map



(c)3D 点云地图

(c)3D point cloud map

图 6-10 栅格地图和点云地图

Fig.6-10 Grid map and point cloud map

6.3 本章小结

本章有介绍了实验环境场景以及实验平台配置，详细描述了利用标准测试数据集和实际场景对本文的视觉RGB-D SLAM算法进行了测试与评估，最后详细叙述了移动机器人基于视觉RGB-D SLAM算法进行自主探测室内实际场景实验。最终结果表明了本文设计的是视觉RGB-D SLAM算法具有实用性和有效性。

总结与展望

SLAM 技术是实现移动机器人自主导航的关键技术之一，由于视觉传感器具有便宜、丰富信息量、轻巧等诸多优点，视觉 SLAM 的研究是移动机器人视觉导航中的核心问题之一。本文介绍了移动机器人视觉导航的研究历程，以及移动机器人 SLAM 问题的研究背景。本文利用 RGB-D 视觉传感器实现了一种视觉 SLAM 算法，有效地实现了移动机器人的即时定位与地图构建。本文完成的主要工作总结如下：

1、查阅大量国内外文献资料，分析了移动机器人视觉导航中的视觉 SLAM 问题，阐述了图像匹配技术和三维重建的研究现状，以及概述了视觉 SLAM 研究现状。

2、获取三维信息，重建三维点云地图。本文使用的 RGB-D 传感器是华硕 Xtion PRO 相机，讲述了 Xtion PRO 相机获取 RGB 图像数据和深度图像数据的实现步骤，以及利用双边滤波预处理对深度图像进行降噪和填补空洞。利用最小化残差平方得到相邻点云数据的相对位置关系，进而对点云数据进行配准与融合，实现了重建三维点云地图。

3、视觉 SLAM 前端一般需实现特征提取与描述子计算、特征匹配与运动估计。比较了 SIFT、SURF 和 ORB 这三种特征点算法的提取特征点数目、运算时间以及正确率，本文最终使用了 ORB 算法实现特征提取与描述子计算。利用 RANSAC 对相邻图像进行匹配，使得匹配图像的相同特征点间能够建立正确的对应关系。描述了根据图像间特征点对应的三维匹配点对集，利用 RANSAC 算法初步求解出相机运动估计，进而利用 ICP 算法优化位姿转换关系。

4、实现视觉 SLAM 后端的方法主要有基于滤波器和基于图优化，详细介绍了基于基于滤波器方法的思想和实现流程，而基于滤波器的视觉 SLAM 存在难以建立大型环境的地图等不足。本文实现了基于图优化的视觉 SLAM 后端，构建了 SLAM 图优化模型，对基于图优化的 SLAM 问题进行了数学推导，并使用 g2o 开源库来实现了图优化。最后利用 BOVW 闭环检测算法提高了移动机器人在未知场景环境下对突发状况的鲁棒性以及重定位。

5、本文提出了一种基于边界的探测策略来实现移动机器人自主探测未知环境并构建 2D 导航地图和 3D 点云地图，采用基于 $\mathbf{A}^*$ 算法的全局路径规划搜索最短行走路径和基于动态窗口法的局部路径规划进行实时避障。

6、不仅利用标准测试数据集和实际场景对本文的视觉 RGB-D SLAM 算法的各个

阶段进行了测试，而且对移动机器人基于视觉 RGB-D SLAM 算法进行自主探测室内实际场景的实验，对实验数据进行整理与分析。

在本论文中，仍有许多问题有待于进一步研究与完善，具体包括以下几个方面：

1、在特征提取与描述子计算方面尝试使用更新更快的算法或者基于 GPU 的方法，进一步提高整体 SLAM 算法的效率。

2、在三维地图重建方面，尝试对地图重建过程进行优化，进而降低系统算法的运算量。

3、本文采用的路径规划算法都属于经典算法，后续尝试对路径规划算法加以改进，提高计算效率。

4、将视觉、激光和 IMU 三者数据融合，以及对图优化的位姿图构建方法和闭环检测的约束条件进行改进与优化，使本文的 SLAM 算法能适用于室外大尺度环境场景。

参考文献

[1] Nilsson N J. A Mobius Automation:an Application of Artificial Intelligence Techniques[C]//Proceedings of the 1st international joint conference on Artificial intelligence. San Francisco:Morgan Kaufinann Publishers Inc, 1969:509-520.

[2] Joseph L J. Robots at the tipping point:the road to irobot roomba[J]. IEEE Robotics & Automation Magazine, IEEE, 2006. 13(1):76-78.

[3] Jia Y H, Mei F X. Simple Path Planning for Mobile Robots in the Present of Obstacles[M], Journal of Beijing Institute of Technology, 2002.

[4] 王建农, 吴捷. 自主移动机器人的导航研究[R]. 机器人, 1997. 19(6): 461-473.

[5] 朴松昊, 洪炳熔. 一种动态环境下移动机器人的路径规划方法.机器人[R], 2003. 25(l):18-19.

[6] 罗荣华, 洪炳熔. 移动机器人同时定位与地图创建研究进展[J]. 机器人, 2004, 26(2):183-186.

[7] Leonard J J, Durrant-Whyte H F, Cox I J. Dynamic Map Building for an Autonomous Mobile Robot[J]. International Journal of Robotics Research, 1990, 11(4):286-298.

[8] Hogman V. Building a 3D Map from RGB-D sensors[D]. Stockholm, Sweden: Computer Vision and Active Perception Laboratory, Royal Institute of Technology (KTH). 2013.

[9] Smith R, Self M, Cheeseman P. Estimating Uncertain Spatial Relationships in Robotics[C]//Proceedings of the IEEE International Conference on Robotics and Automation. IEEE, 1987:850-850.

[10] Leonard J J, Durrant-Whyte H F. Mobile robot localization by tracking geometric beacons[J]. IEEE Transactions on Robotics and Automation, 7(4):376-382, 1991.

[11] Montemerlo M, Thrun S, Koller D, et al. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges[C]//International Joint Conference on Artificial Intelligence. Morgan Kaufmann, 2003, 133(1):1151-1156.

[12] Lu F, Milios E. Globally consistent range scan alignment for environment mapping[J].

Autonomous Robots, 1997, 4(4):333-349.

[13] Thrun S, Montemerlo M. The graph SLAM algorithm with applications to large-scale mapping of urban structures[J]. The International Journal of Robotics Research. 2006, 25(5-6): 403-429.

[14] Li Y, Olson E B. Extracting general-purpose features from LIDAR data[C]. IEEE,2010:1388-1393.

[15] Fulton R R, Hutton B F, Braun M, et al. Use of 3D reconstruction to correct for patient motion in SPECT[J]. Physics in Medicine & Biology, 1994, 39(3):563-74.

[16] 杨鸿, 钱堃, 戴先中, 等. 基于 Kinect 传感器的移动机器人室内环境三维地图创建[J]. 东南大学学报(自然科学版), 2013, 43(A01):183-187.

[17] 贾松敏, 王可, 郭兵, 等. 基于 RGB-D 相机的移动机器人三维 SLAM[J]. 华中科技大学学报(自然科学版), 2014(1):103-109.

[18] 梅峰, 刘京, 李淳秡, 等. 基于 RGB-D 深度相机的室内场景重建[J]. 中国图象图形学报, 2015, 20(10):1366-1373.

[19] Henry P, Krainin M, Herbst E, et al. RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments[J]. International Journal of Robotics Research, 2012, 31(5):647-663.

[20] Engelhard N, Endres F, Hess J. Real-time 3D visual SLAM with a hand-held RGB-D camera[C]//In Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum, 2011 (2011).

[21] Cui Y, Chang W, Li T, et al. KinectAvatar: fully automatic body capture using a single kinect[C]//International Conference on Computer Vision. IEEE, 2012, 7729:133-147.

[22] Newcombe R A, Izadi S, Hilliges O. KinectFusion: Real-time dense surface mapping and tracking[C]//IEEE International Symposium on Mixed & Augmented Reality. IEEE, 2012:127-136.

[23] Kainz B, Hauswiesner S, Reitmayr G, et al. OmniKinect: real-time dense volumetric data acquisition and applications[C]//Acm Symposium on Virtual Reality Software & Technology. ACM, 2012:25-32.

[24] Alexiadis D S, Zarpalas D, Daras P. Real-Time, Full 3-D Reconstruction of Moving Foreground Objects From Multiple Consumer Depth Cameras[J].IEEE Transactions on

Multimedia, 2013, 15(2):339-358.

[25]陈伟, 吴涛, 李政, 等. 基于粒子滤波的单目视觉 SLAM 算法[J]. 机器人, 2008, 30(3):242-247.

[26]Smith R, Self M, Cheeseman P. Estimating uncertain spatial relationships in robotics[C]//Uai 86: Second Conference on Uncertainty in Artificial Intelligence. Elsevier, 1986, 5(5):435-461.

[27]Chiuso A , Favaro P, Jin H, et al. 3-D Motion and Structure from 2-D Motion Causally Integrated over Time: Implementation[M]. Computer Vision ECCV 2000. Springer Berlin Heidelberg, 2000, 24(4):523-535.

[28]Einicke G A, White L B. Robust extended Kalman filtering[J]. Signal Processing IEEE Transactions on, 1999, 47(9):2596-2599.

[29]Montemerlo M, Thrun S, Koller D, et al. FastSLAM:a factored solution to the simultaneous localization and mapping problem[J]. Archives of Environmental Contamination & Toxicology, 2003, 50(2):240-248.

[30]Montemerlo M, Thrun S, Roller D, et al. FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges[C]//International Joint Conference on Artificial Intelligence. Morgan Kaufmann, 2003, 133(1):1151-1156.

[31]Grisettiyz G, Stachniss C, Burgard W. Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling[C]//IEEE International Conference on Robotics & Automation. IEEE, 2005, 312(20):2432-2437.

[32]Sibley G, Matthies L, Sukhatme G. Sliding window filter with application to planetary landing[J]. Journal of Field Robotics, 2010, 27(5):587-608.

[33]康轶非, 宋永端, 宋宇, 等. 平方根容积卡尔曼滤波在移动机器人 SLAM 中的应用[J]. 机器人, 2013, 35(2):186-193.

[34]宋宇, 李庆玲, 康轶非, 等. 平方根容积 Rao-Blackwillised 粒子滤波 SLAM 算法[J]. Acta Automatica Sinica, 2014, 40(2):357-367.

[35]Strasdat H, Montiel J M M, Davison A J. Visual SLAM: Why filter?[J]. Image & Vision Computing, 2012, 30(2):65-77.

[36]Lu F, Milios E. Globally Consistent Range Scan Alignment for Environment Mapping[J]. Autonomous Robots, 1997, 4(4):333-349.

[37]Kaess M, Ranganathan A, Dellaert F. iSAM: Incremental Smoothing and Mapping[J]. IEEE Transactions on Robotics, 2008, 24(6):1365-1378.

[38]Klein G, Murray D. Parallel Tracking and Mapping for Small AR Workspaces[C]//IEEE & Acm International Symposium on Mixed & Augmented Reality. IEEE, 2007:1-10.

[39]Mur-Artal R, Montiel J, Tardós J. ORB-SLAM: A Versatile and Accurate Monocular SLAM System[J]. IEEE Transactions on Robotics, 2015, 31(5):1147-1163.

[40]Gokturk SB, Yalcin H, Bamji C. A Time-Of-Flight Depth Sensor - System Description, Issues and Solutions[C]//Conference on Computer Vision & Pattern Recognition Workshop. IEEE, 2004:35-35.

[41]Stoyanov T, Mojtahedzadeh R, Andreasson H, et al. Comparative evaluation of range sensor accuracy for indoor mobile robotics and automated logistics applications[J]. Robotics & Autonomous Systems, 2013, 61(10):1094-1105.

[42]冯焕飞. 三维重建中的相机标定方法研究[D]. 重庆:重庆交通大学, 2013.

[43]朱德海. 点云库 PCL 学习教程[M]. 北京航空航天出版社, 2012:34-38.

[44]Nock C A, Olivier T, Sylvain D, et al. Assessing the Potential of Low-Cost 3D Cameras for the Rapid Measurement of Plant Woody Structure[J]. Sensors, 2013, 13(12):16216-16233.

[45]Tomasi C, Manduchi R. Bilateral filtering for gray and color images[C]//International Conference on Computer Vision. IEEE, 1998:839 - 846.

[46]张育锋. 三维数据点云的去噪及其检测[D]. 南京:南京信息工程大学, 2014.

[47]Paris S, Durand F. A Fast Approximation of the Bilateral Filter Using a Signal Processing Approach[J]. International Journal of Computer Vision, 2009, 81(1):24-52.

[48]Lowe D G. Distinctive Image Features from Scale-Invariant Keypoints[J]. International Journal of Computer Vision, 2004, 60(2):91-110.

[49]Bay H, Ess A, Tuytelaars T, et al. Speeded-Up Robust Features (SURF)[J]. Computer Vision & Image Understanding, 2008, 110(3):346-359.

[50]Calonder M, Lepetit V,Rublee E, Rabaud V, Konolige K, et al. ORB: An efficient alternative to SIFT or SURF[C]//IEEE International Conference on Computer Vision.

IEEE, 2011, 58(11):2564-2571.

[51] Strecha C, et al. BRIEF: Binary Robust Independent Elementary Features[C]//European Conference on Computer Vision. Springer, 2010, 6314:778-792.

[52] Rosten E, Drummond T. Machine learning for high-speed corner detection[C]//European Conference on Computer Vision. Springer, 2006, 3951:430-443.

[53] 薛振华. 图像局部不变性特征提取与匹配[D]. 天津:天津大学, 2013.

[54] 宋卫艳. RANSAC 算法及其在遥感图像处理中的应用[D]. 北京:华北电力大学, 2011.

[55] Rusinkiewicz S, Levoy M. Efficient variants of the ICP algorithm[C]//The Third International Conference on 3-D Digital Imaging and Modeling. IEEE, 2001: 145-152.

[56] Kalman R E. A New Approach to Linear Filtering and Prediction Problems[J]. Journal of basic Engineer, 1960, 82(1):35-45.

[57] 王开宇, 夏桂华, 朱齐丹, 等. 基于 EKF 的全景视觉机器人 SLAM 算法[J]. 计算机应用研究, 2013, 30(11):3320-3323.

[58] Kummerle R, Grisetti G, Strasdat H, et al. G2o: A general framework for graph optimization[C]//IEEE International Conference on Robotics and Automation. IEEE, 2011:3607-3613.

[59] González-Baños H H, Latombe J C. Navigation Strategies for Exploring Indoor Environments[J]. International Journal of Robotics Research, 2002, 21(10):829-848.

[60] Ganganath N, Leung H. Mobile robot localization using odometry and kinect sensor[C]//IEEE International Conference on Emerging Signal Processing Applications. IEEE, 2012:91-94.

[61] 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 & Cybernetics, 1968, 4(2):100-107.

[62] Khatib O. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots[J]. International Journal of Robotics Research, 1986, 5(1):90-98.

[63] Goldberg D E, Holland J H. Genetic Algorithms and Machine Learning[J]. Machine Learning, 1988, 3(2):95-99.

[64] Liu N H, Hsu H M, Huang T C. Enhanced Approaches for Processing Window Queries in Dynamic Wireless Sensor Networks[J]. Applied Mechanics & Materials, 2011, 58-60:2122-2127.

[65] http://vision.in.tum.de/data/datasets/rgbd-dataset.

攻读学位期间发表的论文及申请的专利

[1]池鹏可，苏成悦，谢广泉，等. Android 平台上的多用户无线投影控制系统设计[J]. 计算机工程与科学. 2015, 37(11): 2148-2153.

[2]池鹏可，苏成悦. 移动机器人中视觉里程计的研究[J].广东工业大学学报(已录用).

学位论文独创性声明

学位论文独创性声明

本人郑重声明：所呈交的学位论文是我个人在导师的指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其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任何贡献均已在论文中作了明确的说明，并表示了谢意。本人依法享有和承担由此论文所产生的权利和责任。

论文作者签名：池鹏宇 日期：2017.6.6

学位论文版权使用授权声明

本学位论文作者完全了解学校有关保存、使用学位论文的规定，同意授权广东工业大学保留并向国家有关部门或机构送交该论文的印刷本和电子版本，允许该论文被查阅和借阅。同意授权广东工业大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印、扫描或数字化等其他复制手段保存和汇编本学位论文。保密论文在解密后遵守此规定。

论文作者签名：池鹏宇 日期：2017.6.6

指导教师签名：[illegible] 日期：2017.6.6

69

致 谢

岁月如梭，如歌。转眼间，三年的研究生求学生活即将结束，站在毕业的门槛上，回首往昔，奋斗和辛劳成为丝丝的记忆，甜美与欢笑也都尘埃落定。广东工业大学以其优良的学习风气、严谨的科研氛围教我求学，以其博大包容的情怀胸襟、浪漫充实的校园生活育我成人。

本论文是在导师苏成悦教授的悉心指导之下完成的。三年来，导师渊博的专业知识，严谨的治学态度，精益求精的工作作风，诲人不倦的高尚师德，朴实无华、平易近人的人格魅力对我影响深远。导师不仅授我以文，而且教我做人，虽历时三载，却赋予我终生受益无穷之道。本论文从选题到完成，几易其稿，每一步都是在导师的指导下完成的，倾注了导师大量的心血，在此我向我的导师苏成悦教授表示深切的谢意与祝福！

另外感谢课题组成员林上飞、林君宇、杨孟军、肖志聪、张洁鑫、张勇、王木华对我在学习和科研上的帮助。有了他们的帮助，我的研究才能够顺利的完成，和他们相处的日子里也让我感受到团队的温馨。

还要感谢父母的养育之恩，以及在我求学生涯中给与我无微不至的关怀和照顾，一如既往地支持我、鼓励我，是他们近 20 多年来精神上和经济上的全力支持才使得我能够完成最后的学业，对此是万分敬意和感谢。

最后，值此毕业论文完成之际，我谨向所有关心、爱护、帮助我的人们表示最诚挚的感谢与最美好的祝愿。