电子科技大学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

硕士学位论文

MASTER THESIS

论文题目 基于三维视觉的 SLAM 技术研究

学 科 专 业 计算机应用技术

学　　号 201521240302

作 者 姓 名 曾创

指 导 教 师 于鸿洋　副教授

万方数据

分类号 _____ 密级 _____

UDC [注1] _____

学 位 论 文

基于三维视觉的 SLAM 技术研究

（题名和副题名）

曾创

（作者姓名）

指导教师 **于鸿洋** **副教授**

电子科技大学 **成都**

（姓名、职称、单位名称）

申请学位级别 **硕士** 学科专业 **计算机应用技术**

提交论文日期 2018-5-29 论文答辩日期 2018-5-30

学位授予单位和日期 **电子科技大学** 2018-6-26

答辩委员会主席 _____

评阅人 _____

注 1：注明《国际十进分类法 UDC》的类号。

万方数据

Research on SLAM Technology Based on 3D Vision

A Master Thesis Submitted to

University of Electronic Science and Technology of China

Discipline: **Technology of Computer Application**

Author: **Zeng Chuang**

Supervisor: **Associate Prof. Yu Hongyang**

School: **Research Institute Electronic Science and Technology of UESTC**

万方数据

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，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，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。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谢意。

作者签名：_____ 日期：2018 年 5 月 29 日

论文使用授权

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

作者签名：_____ 导师签名：_____

日期：2018 年 5 月 29 日

摘要

基于三维视觉的 SLAM 技术采用双目或者 RGBD 相机拍摄的图像为数据输入，通过 SLAM 技术进行场景地图创建，同时让机器人实时定位。现如今同时定位与地图创建技术发展迅速，在无人驾驶，无人机等方面广泛应用。本文研究重点包括设计一种更高效和鲁棒性的视觉特征提取与匹配算法，以及非线性 SLAM 优化算法，回环检测算法等关键技术。本文主要内容如下：

1、本文对图像特征的提取与匹配算法进行了重点研究，在传统的 SLAM 视觉匹配算法基础上，新增了贝叶斯网格统计算法。本文利用经典特征提取描述子和贝叶斯网格统计，可以在不失去精度的情况下，加速特征点的匹配，错误匹配点的剔除。实验结果表明融入了贝叶斯网格统计算法后的 SLAM 框架能够比传统框架达到更佳的计算效率和鲁棒性。

2、本文改进了三维视觉 SLAM 技术中的状态估计算法，对常见的状态估计模型做了量化分析，并指出各自的优缺点。在实际应用过程中，本文采用了光速平差法，通过使用全局时间路标优化了视觉里程计的状态信息，改进了传统的卡尔曼滤波算法。实验结果表明基于光速平差法的状态估计出色的解决了真实场景给 SLAM 带来的非线性影响，同时兼顾了性能和计算复杂度两个方面。

3、本文重构了回环检测算法。在本文的三维视觉 SLAM 技术方案中，贝叶斯网格统计算法会获取大量的特征点。为了达到工程的实时性和最优性，本文在传统的回环检测算法中，新增信息熵率校验环节，通过调整关键帧的提取步骤，快速过滤掉干扰的图像帧，最后获得三维场景的关键帧。提取关键帧后，本文利用无监督学习的方法构建出关键帧的词袋。实验结果验证了重构后的回环检测算法的合理性，新的回环检测算法适用于本文的 SLAM 技术方案，同时也保持了同传统算法模型一致的准确度。

4、本文完成了三维地图创建，将本文算法方案和经典的 SLAM 技术方案进行了一系列的测试对比，验证了本文方案的可行性和准确性。在测试对比中，本文在各个算法时间环节都得到了提升，同时鲁棒性也得到了提高。本文方案不足的地方为执行程序时占用的内存较大。随着时代的发展，内存占用问题可以用硬件逐一解决。

关键词：三维视觉，SLAM，特征匹配，非线性优化，回环检测

万方数据

ABSTRACT

The SLAM technology based on 3D vision uses images captured by binocular or RGBD cameras as data input. The SLAM technology is used to create the scene map, and the robot is positioned in real time. At the same time, simultaneous positioning and map creation technologies have developed rapidly and are widely used in unmanned and unmanned vehicles. The research focus of this thesis includes designing a more efficient and robust visual feature extraction and matching algorithm, as well as key techniques such as nonlinear optimization algorithm and map loop detection algorithm. The main content of this article is as follows:

1. This thesis focuses on the extraction and matching of image features. A Bayesian grid statistical visual feature module is integrated into the traditional SLAM technology framework. Using classical feature extraction descriptors and Bayesian grid statistics, the matching of feature points and the elimination of false matching points can be accelerated without losing precision. Subsequent experiments show that the SLAM framework of the Bayesian grid statistics algorithm module can achieve higher speed and more robustness than the traditional framework.

2. This thesis improves the posture state estimation algorithm for three-dimensional visual SLAM, makes a quantitative analysis of common state estimation models, and points out their respective advantages and disadvantages. In the actual application process, it was found that SLAM has nonlinear interference in the real environment. In this paper, the traditional Kalman filter algorithm is improved and the state estimation and optimization based on the optical speed adjustment method are used. Under certain circumstances, posture state estimation can be performed well. The experimental results show that both the performance and computational complexity are taken into account based on the optical speed adjustment method, and the nonlinear effect of the real scene on the SLAM is excellently achieved.

3. This thesis reconstructs the loop detection algorithm. In the SLAM scheme, the front-end adopts the Bayesian grid visual model to obtain a large number of feature points. In order to achieve real-time and engineering optimality, an addition and subtraction method is applied to the traditional loopback detection algorithm. In the loop detection

万方数据

algorithm, a new information entropy rate verification link is added, which can quickly filter and obtain key frames. Finally, the unsupervised learning method is used to construct the keyword frame of key frames. The new loopback detection algorithm is applicable to one of the algorithm schemes in this paper. The rationality of the loopback detection algorithm after reconstruction is verified through experiments, and the new algorithm scheme also maintains the same accuracy as the traditional algorithm model.

4. The thesis has completed the creation of 3D maps. A series of tests and comparisons have been made between the proposed algorithm and the classic SLAM technology, which verifies the feasibility and accuracy of the proposed scheme. In the test comparison, this article has been improved in all aspects of the algorithm time, and the robustness has also been improved, the disadvantage is that the memory consumption of this algorithm is the largest, but with the development of the era of memory occupation can be completely used the hardware solves one by one.

Keywords: Three-dimensional vision, SLAM, Feature matching, Nonlinear optimization, Loopback detection

万方数据

目录

万方数据

万方数据

万方数据

第一章 绪论

1.1 研究背景和意义

随着人工智能应用的广泛发展，工业 4.0 时代里，具备自规划，自适应能力的设备成为人们关注的焦点。人们越来越注重机器人的自主化和智能化水平。自主移动机器人已经在自动驾驶，无人机，无人零售等领域迅速崛起，并逐渐成为这个世界的一部分，发挥着智能时代新作用。

移动机器人的定位和地图创建(SLAM, simultaneous localization and mapping)是迈向高智能水平的热点研究问题。在一个未知的环境中，机器人需要自己去探索，观察，从而对自身运动进行估计，确定自己的位置，建立环境地图。更进一步通过建立好的环境地图，利用机器学习技术去认知周围的事物，导航，避障，实时提出决策。

目前 SLAM 技术呈现快速发展的态势，2015 年，美国机器人公司率先推出了结合摄像头的扫地机器人，利用轮式里程计使得它能够在较大的空间进行自主定位与建图。2016 年，特斯拉以摄像头为基准，发布了新一代自动驾驶系统 Autopilot2.0，这是首次以大规模视觉提取来对环境进行感知的方案。2018 年，百度提出了无人车量产化的目标，同年在北京进行了无人车的测试，也标志着中国 SLAM 技术的崛起。虽然 SLAM 技术发展迅速，但是测量的不确定性和计算效率问题依旧存在，2018 年 3 月，美国 Uber 无人驾驶测试致人死亡震惊世界。这个事件也让研究人员进一步思考 SLAM 技术中需要改进的地方。

SLAM 技术的问题主要是测量的不确定性，而测量的不确定性主要来源于传感器的噪声或技术的局限性。单一传感器无法在各种应用环境中获得鲁棒性较好的特征提取，比如在城市复杂环境中,激光雷达只能在稳定环境区域里获得稳定的边缘特征，而类似越野环境这种不稳定区域，激光雷达提取到稳定的几何特征将十分困难。而视觉传感器则在不稳定环境区域中能够获得大量特征，增强测量的确定性。所以利用计算机三维视觉可以让整个 SLAM 工程增加更多的高智能性和高精确性。所以越来越多的专家开始研究融入视觉的 SLAM 技术的计算效率和鲁棒性。基于三维视觉的 SLAM 技术就像是赋予了机器人眼睛，让机器人更加清楚，任务规划，最佳路径选择。本文主要研究基于三维视觉的移动机器人同时定位与地图创建相关问题。

万方数据

1.2 国内外研究现状

SLAM 技术是指同时定位与地图创建，人们最开始只是想实现机器人能够读懂的地图来实现定位的一种功能，后来慢慢发展为实时地图创建与定位。最早的地图创建应该是栅格地图，后来又演变为了特征地图和拓扑地图。20 世纪 90 年代的时候，H.P Moravec 和 A.Elfes 提出了第一种描述式环境地图。这是栅格地图的开始，同时还提出了避障的概念，采用概率统计值对地图上每一个网格区域进行部署，用 0-1 来表示有无障碍[1]。后来科研人员开始琢磨使用点线结合方法开始描述环境组建特征地图[2]。由于点线式地图的存储极其复杂，科研人员将各节点用一定方式组织成为一个图结构，而这个图形结构被称为拓扑地图[3]。

Simultaneous Localization and Mapping 一词最开始出现在美国机器人学家 Self 发表的论文中[4][5]。90 年代，SLAM 进程中遇到了很多问题，比如位姿估计。20 世纪末，图优化实现了重大的突破，Lu 巧妙的思考出使用最大似然估计来进行位姿估计，初步解决了 SLAM 中位姿估计的问题。但是当时这系列的求解极其复杂，无法用在实时的场景中[6]。直到研究人员发现非线性最小二乘法的求解，才解决了高维矩阵求逆的问题。

对于视觉 SLAM 的研究一开始来源于视觉里程计的研究，视觉里程计对于相机数量的使用并没有太多规范，单个相机或者组合式相机成为了当时估计视觉轨迹的主流图像输入方式[7]。美国 NASA 部门更是运用位姿估计对火星着陆器实时修正运动倾斜角，来预防打滑的问题[8][9]。

目前为止关于 SLAM 的框架层出不穷，在 2004 年的机器人会议上，Nistér D, Naroditsky O 提出了一种实时的视觉里程计，这成为了后来基于视觉 SLAM 的标准框架[10]。到了 2007 年，对于相机的使用也开始规范为了单目相机或者双目相机，麻省理工的研究人员 Davison 完成了首个单目相机的实时视觉 SLAM 系统框架[5]。基于双目相机的 SLAM 技术计算的运动图像之间的关系，以及扩展卡尔曼滤波的使用，让三维稀疏环境地图的构建更加准确。也就是这个时候，视觉深度在机器人的位姿上得到了更多的应用，地图创建开始有了迭代更新的概念。

环境噪音一直阻碍着视觉 SLAM 的发展，著名的美国教授 Pupilli 使用了一种粒子滤波的方式来逼近噪音的一个分布，逐步在噪音干扰下解决了非线性噪音问题。现在这种概率逼近分布更多用在了激光雷达的 SLAM 上[11][12]，

三维视觉 SLAM 中状态估计尤为重要，状态估计一般使用的是卡尔曼滤波器。但是卡尔曼滤波的只能让视觉里程计在局部领域内够有较好的效果，使得应用环境的范围受到限制。美国著名的数学研究人员从 EFK 本质原理入手，发现了 EFK 出现的问题所在，如果状态变量仅仅是使用现在时刻的位姿，而不将过去的位姿进

万方数据

行同步更新，那么添加到 EFK 的变量会出现累积误差，导致整个环境地图的扩增[13]。

到了 2007 年，Mourikis 利用多状态约束卡尔曼滤波方法分化了状态估计的过程，提出了状态预测和状态更新，利用滑动窗口对相邻 N 帧相机运动参数进行过滤，大大减少了特征点数量对算法的线性增长性。同时设置了状态估计更新规则，只有满足阈值条件才进行一个运动状态的更新[14]。

到了 2008 年，Mourikis 在原有基础上添加了估计任务量，改善了滤波器的可观测性[15]。如图 1-1，左图：追踪特征点。右图：特征点在三维空间中的表示。

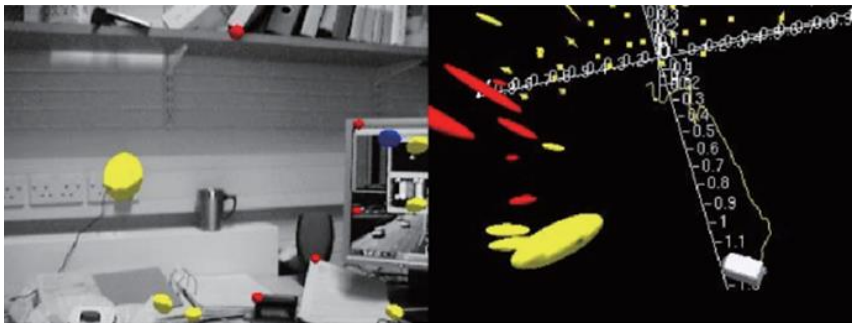


图 1-1 MonoSLAM 的运行时截图

MonoSLAM 在运行时，图 1-1 左图单目相机利用主动跟踪技术，在一幅图像描绘出较多的稀疏特征信息点[15]。图 1-1 右图可以看到空间中的一些小球体，这些球体在某个方向显得比较长。结合 EKF 容易明白，每个特征点的位置信息是服从高斯分布，图中的球体代表了高斯分布的均值以及一些不确定值。右图中有部分小球呈现拉长状，说明均值非常不稳定，那么它的位置信息波动也会很大。从图中可以看出一个特征点进入收敛过程时，球体会从一个椭圆形慢慢的变成小，直至到收敛的最小值形成一个小点。

MonoSLAM 开创了很多东西，首先是真正意义上的在线运行，而不再是靠机器人离线对图像收集，然后再进行地图创建。其次是图像稀疏表达的运用，真正解决了算法计算复杂性的问题。虽然 MonoSLAM 在路标少，场景复杂的环境中，仍然无法达到较好达到效果。但在那个年代，SLAM 的开发已经达到了一个小的顶峰。

到了 2008 年，视觉 SLAM 又出现了 PATM(Parallel Tracking and Mapping)框架。这个框架是 Georg Klein 和 David 设计，主要目的是用在增强现实领域中的[16]。PATM 框架提出了两个线程，一个线程负责实时数据的处理，另一个负责后端地图的优化。PATM 使用的是主流的特征点算法，利用特征点进行图像的实时响应构建相机的位姿轨迹。不同的是 PATM 采用了 Bundle Adjust 进一步解决了 EFK 的局限

万方数据

性。而且 PATM 第一个提出了前后端的概念，让前端集中精力去实时计算关键位姿，让费事的环节比如地图的构建和优化交给后端进行响应反馈。这种做法提高的算法的容错性，虽然在大型场景中仍然出现了问题，但是 SLAM 走向成熟化框架的标志，如图 1-2，PTAM 框架将增强现实和 SLAM 的一次结合。

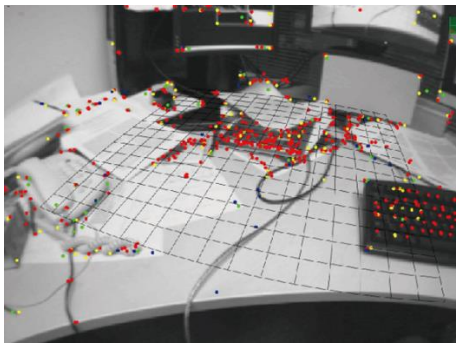


图 1-2 PTAM 的演示截图。

到了 2015 年， 研究人员 Raul 对 PATM 框架进行了一个新的重构，将 PATM 中的算法进行了较多的改进。首先是在 PATM 里面融入了 ORB 算子特征点追踪和匹配[17]，并把这种新的算法方案命名为 ORB-SLAM。同时 ORB-SLAM 在原有 PATM 结构里再次增加了一个线程，这个线程主要负责回环检测。ORB-SLAM 通过回环检测的运行，让系统拥有了更高的定位和地图创建的精确度。除了增加线程，Raul 还修改了部分初始化环节。新的初始环节使用了单应化矩阵以及基础矩阵模型对地图进行自动初始化。ORB-SLAM 的大规模实际应用证明了同时定位和地图创建技术的逐步成熟。ORB-SLAM 的建立让科研人员有了新的目标，那就是开始着手于更高实时和精度的系统。

2015 年后，SLAM 技术研究变得细致起来，研究人员开始对复杂环境下前端视觉里程计特征点提取与匹配进行研究。特别是在缺少图像信息的环境下，科研人员尝试通过滤波或者场景修复来挖掘可用的图像特征。这样可以获得更高的位姿精度，提高地图创建的准确性。LSD-SLAM 是一项较为震惊的工程，它是由 J. Engle 等人在结合不同图像环境下提出的一种新的解决方案[18]。

LSD-SLAM 丰富了人们对地图的需求，它不仅可以跟踪梯度变化最为明显的像素，还能结合逆深度三维参数表达图像内在联系。LSD-SLAM 依靠大量的关键帧来建立稠密地图或者半稠密地图。在不同场景下，LSD-SLAM 能够降低运动漂移的影响，同时位姿估计上真正做到不受尺度干扰。LSD-SLAM 唯一出现使用限制的地方是环境对 SLAM 实体大小的要求。因为稠密地图的创建极其消耗时间，单个 CPU 肯定是无法达到最佳效果，也许 SLAM 的发展也要开始考验硬件上的设计了[19]。如图 1-3 为 LSD-SLAM 半稠密地图。

万方数据

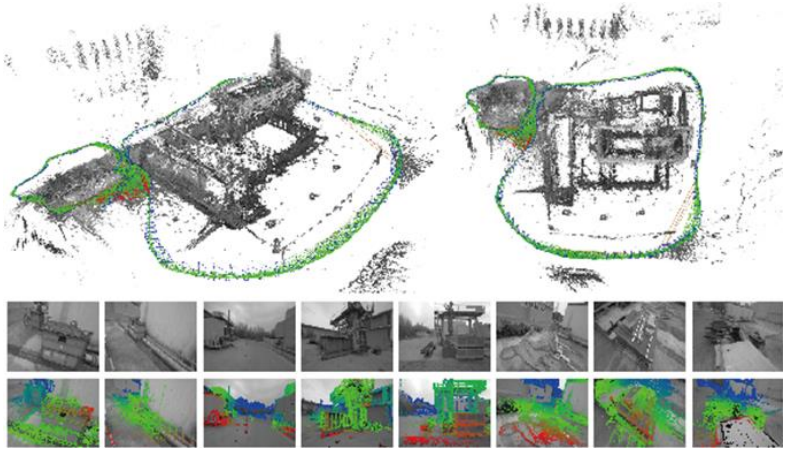


图 1-3 LSD-SLAM 半稠密地图

1.3 论文主要工作和创新点

1.3.1 论文主要工作

本文的主要工作如下：

整理了 SLAM 技术发展过程，梳理了三维视觉 SLAM 整个技术框架，重点分析了三维视觉 SLAM 的几个关键环节。其中发现的问题如下：

在三维视觉 SLAM 算法中，经典的 SIFT 特征提取算法在工程应用上是无法使用的，原因在于 SITF 算法提取的特征点向量维数导致高计算复杂度，严重制约了系统的实时性。目前视觉里程计中的特征提取方法大多数是 ORB 特征提取算法，通过将特征点向量维数降低了一半，虽然达到了实时的使用效果，却因此丢失了部分精度，在质量差和纹理低的场景中，匹配效果较差。如果增加 ORB 的特征提取点数，又会导致后序匹配筛选以及计算运动关系速度降低。

在三维视觉 SLAM 状态估计环节，传统算法使用的是扩展卡尔曼滤波进行最大后验概率估计，问题在于扩展卡尔曼滤波器解决非线性化时的泰勒展开式的第一项，只能用来描绘较近工作点的非线性情况。对于较远工作点，扩展滤波器这种小范围线性近似方法，无法适用于很远的地方，极容易产生非线性误差。

在三维视觉 SLAM 回环检测算法中，第一个问题是感知上的分歧，也就是两个相似的观测信息由于环境光线或者其他因素导致可辨性受到制约，极其容易判断的错误，第二个就是数据规模，随着地图的创建，数据规模也会越来越大，存储计算要求也越来越大。第三个就是对错误的回环检测的修正，不让漂移误差影响后续优化的收敛性，由此回环的匹配准确度也是一个重要的问题。

针对上面三个问题，做出了如下工作：

1、研究了三维视觉 SLAM 的特征和匹配过程中出现复杂度高的算法环节，在

万方数据

传统的视觉提取基础上，通过建立贝叶斯视觉模型，利用网格区域的加权统计，设计出一种贝叶斯网格统计算法模块，加速了视觉特征匹配效率，提高了传统匹配算法的鲁棒性。

2、研究了三维视觉 SLAM 状态估计概率模型，梳理清楚了卡尔曼滤波器内在原理。在马尔科夫假设基础上，利用光速平差法在新的估计点重新进行非线性优化，将一阶或二阶的梯度下降估计灵活运用到新的状态估计点处。光速平差法让整个优化适用范围更广，同时结合矩阵稀疏性，加快了整个状态估计的速度。

3、研究了三维视觉 SLAM 的回环检测算法，利用非监督学习对图像特征点进行单词构建，通过词袋中字典的对比获得回环反馈。本文分析了原有的关键帧提取算法，将算法结构进行了一定调整，增加了信息熵的比较环节，利用信息熵对两帧之间的熵率进行对比，如果两帧之间的信息熵率的比值越大，那么图像灰度分布情况会存在一个较小的差异，对应的两幅图像息也会产生较小的差异。相反，如果两帧之间的信息熵率的比值越小，那么两幅图像的灰度分布会存在较大的差异，对应的两幅图像特征信息会是较大的差异。通过增加信息熵环节的比较加快了词袋单词的比对速度，优化了关键帧提取的效率和准确度。

总的来说，本文主要工作是对三维视觉 SLAM 核心环节进行研究，旨在获得一个最优化的三维视觉 SLAM 框架。通过实验也验证了改进算法的可行性，优化了三维视觉 SLAM 运行速度和鲁棒性。

1.3.2 论文创新点

1、本文设计了一种基于贝叶斯网格统计特征提取匹配算法，在传统 SLAM 技术框架中，融入贝叶斯网格统计算法模块，可以加速运动模型的计算，提高系统的鲁棒性。其核心思想是根据人的视觉猜测建立区域贝叶斯特征支持，利用网格切分和量化统计快速匹配到支持区域的所有特征点。整个贝叶斯网格统计模块在 CPU 运行下只需要 1.5ms。在同样特征点数量提取情况下，比传统的 ORB 提取匹配效率提高了 10%。

2、本文对 SLAM 技术算法中进行了一些调整，不影响效率下，增加了特征点数量，减少了关键帧的数量。同时基于帧间熵率的关键帧检测，让关键帧的提取速度得到了提高。

1.4 论文的结构安排

本文首先介绍基于三维视觉的 SLAM 技术过程中的背景及其研究现状，然后依据三维视觉 SLAM 技术的关键技术点进行分章阐述，最后通过实验验证相关

万方数据

SLAM 算法，并对实验结果进行量化分析。全文章节安排如下：

第一章介绍了课题研究的背景和意义、国内外研究现状、论文的主要工作和创新点、论文的结构安排等内容。

第二章主要研究了三维地图创建和定位的相关原理和更新机制。本章首先介绍了机器人的位姿描述与坐标变换，了解了地图创建过程中的图像之间的运动变换的空间转移关系。本文结合深度信息和彩色信息形成三维点云数据，最后利用相机空间转移关系实时更新地图和定位。

第三章主要梳理了经典的特征提取与匹配算法，本章研究了三维视觉 SLAM 的视觉特征和匹配过程中出现复杂度高的算法环节，在传统的视觉提取基础上，通过建立贝叶斯视觉模型，利用网格区域的加权统计，设计出一种贝叶斯网格统计算法模块，通过实验验证了该算法模块加速了视觉特征匹配效率，提高了传统匹配算法的鲁棒性。

第四章主要讨论了三维视觉 SLAM 的优化算法。本章研究了 SLAM 状态估计的计算以及针对状态估计的优化措施，从状态估计概率模型的建立，核心马尔科夫思想的问题研究。根据前面算法的修改，本章为了达到原本算法的稳定性，将光速平差法与稀疏化矩阵求解加入了了非线性优化计算中，实现了适用于贝叶斯视觉网格的状态优化算法。

第五章主要重构了回环检测算法的流程，本章首先介绍了词袋的构成原理，将原有的基于欧式距离的关键帧算法进行了重构，增加了信息熵初校验，以及贝叶斯视觉网格匹配过滤，完成了整个回环检测的算法流程，同时本章通过实验验证了算法的合理性，取得了一定的优化成果。

第六章主要是一个算法实验的总结分析，本章从各个指标上对比了各个特征提取与匹配算法的差异性，验证了本文设计的贝叶斯网格统计模块的优化性，以及非线性优化算法的改进的合理性。最后本章完成了回环检测的一个算法重构，得到了三维地图的实验结果。

第七章对于本课题所做的研究工作进行了总结，指出了研究过程中存在的不足，并对下一步研究做出展望。

万方数据

第二章 三维地图创建与定位关键技术及原理

本章主要研究了三维地图创建与定位的原理，系统梳理了三维地图创建与定位的整体思路，如图 2-1 所示。首先本文利用三维相机获得输入数据，这个数据来源于室内环境的深度与彩色信息，根据彩色信息和深度信息的配准，可以将其融合成具有三维信息的点云数据。随着机器人的移动，本文提取移动帧的视觉特征，通过视觉特征的匹配计算三维运动估计，得到图像间的变换关系。最后将三维点云按照运动关系进行坐标变换创建出三维地图[24][25]。建立何种地图是根据实际应用来决定，稠密地图需要提取的图像帧数较多，所以这种地图应用范围广，但是存储空间大，计算效率慢。而稀疏地图则在创建的帧数中提取一定量（而不是全部）关键帧，存储空间小，应用范围较广，计算速度快。本文在关键帧的基础上进行回环检测同时解决累积误差漂移的问题。在回环检测中如果发现闭合路径，则动态调整图优化所得的图像帧间的约束关系，也就是改变节点位姿，最后重新应用到地图创建并同时更新地图的显示与定位。

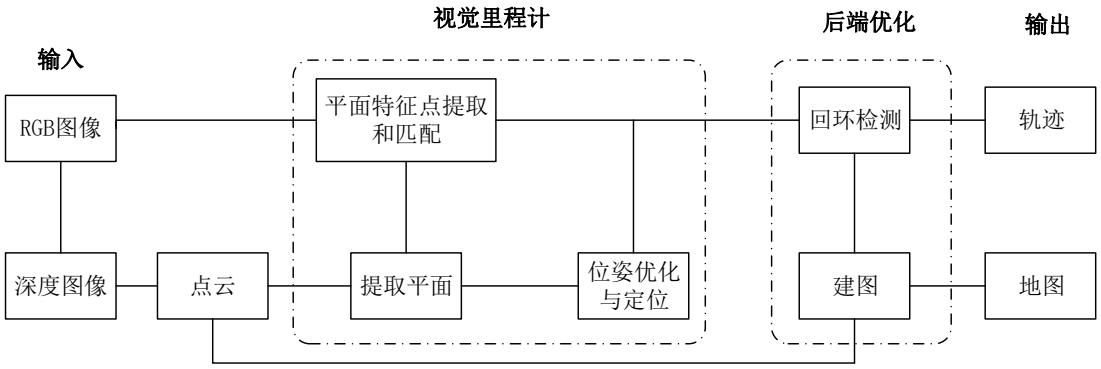


图 2-1 三维地图创建与定位技术框图

三维地图创建与定位技术，输入的信号量往往是一组 RGB 图像和深度图像，利用 RGB 图像可以获取到实时场景的相关特征信息，使用深度数据可以构建场景三维点云图像，为后面的定位与更新地图做好铺垫。

视觉里程计也就是 SLAM 的前端，主要任务是估算相邻图像之间的相机运动，或者说就是计算这种运动之间的数学关系。这种数学关系其实就是相机与空间怎么转换的连接，那么怎么才能计算这种转换呢？当然视觉里程计除了计算空间转换，还能够通过相邻帧间的图像特征点的提取和匹配，深度图像数据对应，恢复场景的空间结构。需要注意的是视觉里程计只是计算相邻时刻视觉图像的运动，和过去的信息没有关系。

万方数据

后端优化的核心其实就是回环检测。视觉里程计估计轨迹，往往不可避免地出现累积漂移。在视觉里程计估计两个图形间的运动过程中，每次状态位姿估计都会出现一定的偏差，这个偏差对于整个系统其实是致命的，试想视觉里程计在运动中把先前时间点的偏差传到了下一时间点，下一个时间点又把偏差传递了下去，那么最后总的偏差无疑是不能接受的，所估计的轨迹也就失去了意义。所以需要一个后端来对这个误差进行实时的反馈，来修成回环路径。

SLAM 的输出应该是一个具有三维结构的地图，当然也可以将三维结构地图压缩为二维地图方便保存。但是核心都在建图这个任务上面，日常生活中我们使用的地图是由城市来构成的，这些城市抽象来说就是路标，那么在 SLAM 中我们也需要去找这些类似“城市”的路标信息，寻找完所有的路标信息，也就得到了我们所谓的地图。所以前面说的视觉里程计事实上就是在构建路标点的位置，并对它们进行优化，所以视觉里程计的重要性不言而喻。

SLAM 技术不仅是一种底层技术，同时还是面向应用层的技术，经常要求 SLAM 在底层的基础上能够带来更多有用的功能。如果上层是机器人，那么应用层的开发者可能希望使用 SLAM 来做全局的定位，并且让机器人在地图中导航——例如扫地机需要完成扫地工作，希望计算一条能够覆盖整张地图的路径。或者，如果上层是一个增强现实设备，那么开发者可能希望将虚拟物体叠加在现实物体之中，特别地，还可能需要处理虚拟物体和真实物体的遮挡关系。

同时应用层面对于定位的需求是相似的，他们希望 SLAM 提供相机或搭载相机的主体的空间位姿信息。而对于地图，则存在着许多不同的需求。在视觉 SLAM 看来，地图创建是服务于定位的；但是在应用层面看来，建图明显还带有许多其他的需求。关于地图的用处，本文大致归纳如下：

1、定位。定位是地图的一个基本功能。在利用前面的视觉里程计，可以利用局部地图来实现定位。在回环检测里，可以用全局的描述子信息，确定机器人的位置。其实每次进行地图创建是一件浪费时间效率的事情，如今定位要求即时性，也就是应该能够让机器人即时就能定位，而不是每次先建图再定位，所以定位与地图建模存储是密不可分的一件事。

2、导航。机器人从任意两个地图点，利用搜索判断寻找最优路径方案的过程。方案制定完毕就开始控制自己运动到目标点的过程。该过程中至少需要知道地图中哪些地方不可通过，而哪些地方是可以通过的。

3、避障。避障也是机器人经常碰到的一个问题。它与导航类似，但更注重局部的、动态的障碍物的处理。同样的，仅有特征点无法判断某个特征点是否为障碍物，所以我们将需要稠密地图。

万方数据

4、重建。有时候希望利用 SLAM 获得周围环境的重建效果，并把它展示给其他人看。这种地图主要用于向人展示，所以我们希望它看上去比较舒服、美观。或者也可以把该地图用于通讯，使其他人能够远程地观看重建得到的三维物体或场景——例如三维的视频通话或者网上购物等等。这种地图亦是稠密的，并且还对它的外观有一些要求。我们可能不满足于稠密点云重建，更希望能够构建带纹理的平面，就像电子游戏中的三维场景那样。

5、交互。交互主要指人与地图之间的互动。例如，在增强现实中会在房间里放置虚拟的物体，并与这些虚拟物体之间有一些互动——比方说我会点击墙面上放着的虚拟网页浏览器来观看视频，或者向墙面投掷物体，希望它们有（虚拟的）物理碰撞。另一方面，机器人应用中也会有与人、与地图之间的交互。

2.1 机器人的位姿描述与坐标变换

视觉 SLAM 中通常采用相机来获取场景信息，通过描述三维世界的点的相互映射关键，来获取机器人的位姿。本节主要研究视觉里程计的路标构成以及有关机器人视觉里程计应用过程中相机和世界空间的一个变换。

一、相机位姿的数学表达

视觉里程计在移动的过程中往往会发生刚体位姿变换，在这个过程中可以用姿态矢量和位置矢量来描述每一个节点的位姿变化。如图 2-2 所示，定义一个坐标系，则在该坐标系下的某点的位姿可以描述为：

$$\{O'\} = \{{}^{O'}_{O}R, {}^{O'}_{O}P\} \tag{2-1}$$

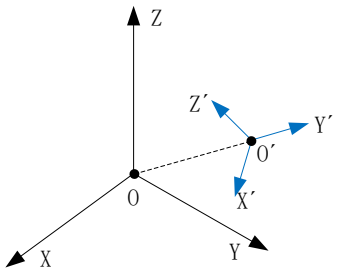


图 2-2 用位置矢量和姿态矢量描述节点

其中 ${}^{O'}_{O}P$ 为位置矢量，如式 2-2：

$${}^{O'}_{O}P = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \tag{2-2}$$

万方数据

其中 ${}^{O'}_{O}R$ 为姿态矢量，如式 2-3：

$$ {}^{O'}_{O}R=\left[\begin{matrix}{}^{O'}_{O}\mathrm{X} & {}^{O'}_{O}\mathrm{Y} & {}^{O'}_{O}Z\end{matrix}\right]_{3\times3}=\left[\begin{matrix}\cos(\angle X'X) & \cos(\angle Y'X) & \cos(\angle Z'X)\\ \cos(\angle X'Y) & \cos(\angle Y'Y) & \cos(\angle Z'Y)\\ \cos(\angle X'Z) & \cos(\angle Y'Z) & \cos(\angle Z'Z)\end{matrix}\right] \tag{2-3} $$

公式 2-3 中，还需要增添 4 个限制条件，如下：

$$ ({}^{O'}_{O}X)^2=({}^{O'}_{O}Y)^2=({}^{O'}_{O}Z)^2=1 \tag{2-4} $$

$$ {}^{O'}_{O}X\bullet{}^{O'}_{O}Y={}^{O'}_{O}Y\bullet{}^{O'}_{O}Z={}^{O'}_{O}X\bullet{}^{O'}_{O}Z=0 \tag{2-5} $$

$$ {}^{O'}_{O}R^{-1}={}^{O'}_{O}R^{T} \tag{2-6} $$

$$ \left|{}^{O'}_{O}R\right|=1 \tag{2-7} $$

二、坐标变换

移动机器人在移动的时候，通过相机获取到环境信息。环境信息的变化可以描述为从相机坐标系变换到所处环境的坐标系。这种大级别的坐标转换包含有三种内部小级别的变换。

1、坐标平面移动

坐标平面移动变换其实就是从一个坐标轴按照某个平面平移到另外一个坐标轴。在整个移动过程里面机器人不涉及到三维同时变化，而只是朝着一个平面轴的变化。坐标平面平移变换如图 2-3：

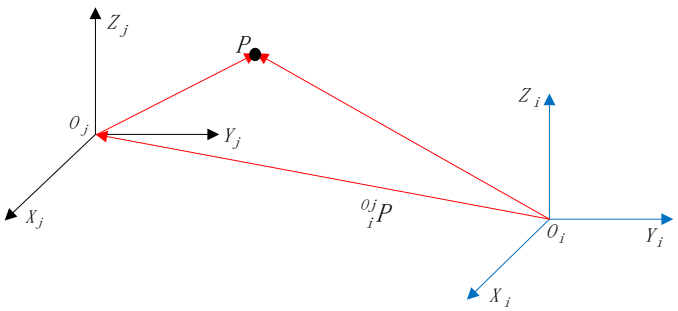


图 2-3 坐标平移变换

例如将从坐标原点分别沿着 x 轴，y 轴，z 轴进行平移，则整个移动过程的关系 ${}^{oj}_{i}P$ 可分解为如式 2-8：

$$ {}^{oj}_{i}P=\left[\begin{matrix}\sum\Delta x\\ 0\\ 0\end{matrix}\right]+\left[\begin{matrix}0\\ \sum\Delta y\\ 0\end{matrix}\right]+\left[\begin{matrix}0\\ 0\\ \sum\Delta z\end{matrix}\right]=\left[\begin{matrix}\sum\Delta x\\ \sum\Delta y\\ \sum\Delta z\end{matrix}\right] \tag{2-8} $$

万方数据

2、坐标轴旋转变换

坐标轴旋转变换就是以坐标原点为轴心，根据不同坐标轴为旋转方向进行旋转运动。例如我们将坐标系 i 旋转到坐标系 j，此时将坐标系 j 下的点 P_j 还原回坐标系下的点 P_i 的过程如图 2-4 所示：

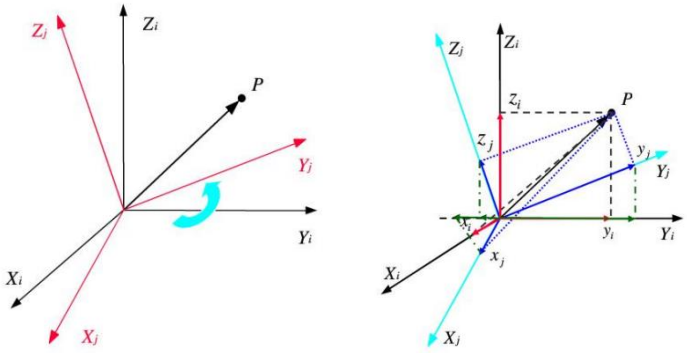


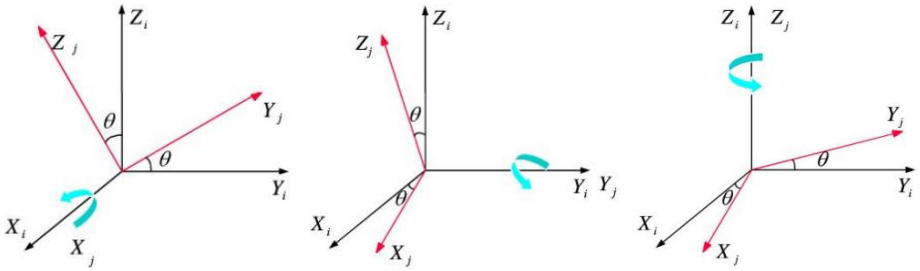
图 2-4 坐标旋转变换

$$P_i=\begin{cases}x_i=x_j\cos(\angle X_i,X_j)+y_i\cos(\angle X_i,Y_j)+z_j\cos(\angle X_i,Z_j)\\ y_i=x_j\cos(\angle Y_i,X_j)+y_i\cos(\angle Y_i,Y_j)+z_j\cos(\angle X_i,Z_j)\\ z_i=x_j\cos(\angle Z_i,X_j)+y_i\cos(\angle Z_i,Y_j)+z_j\cos(\angle Z_i,Z_j)\end{cases} \tag{2-9}$$

$$P_i=\begin{bmatrix}\cos(\angle X'X) & \cos(\angle Y'X) & \cos(\angle Z'X)\\ \cos(\angle X'Y) & \cos(\angle Y'Y) & \cos(\angle Z'Y)\\ \cos(\angle X'Z) & \cos(\angle Y'Z) & \cos(\angle Z'Z)\end{bmatrix}\begin{bmatrix}x_j\\ y_j\\ z_j\end{bmatrix}={}_i^jRP_j \tag{2-10}$$

其中 ${}_i^jR$ 为旋转矩阵，且有 ${}_i^jR={}_i^jR^{-1}={}_i^jR^T$。

从上述原理描述，我们能够分别计算出绕 x 轴，y 轴，z 轴的一个旋转矩阵。如图 2-5 所示。对应的旋转矩阵为式 2-11，式 2-12，式 2-13。



(a) 绕 x 轴旋转　　(b) 绕 y 轴旋转　　(c) 绕 z 轴旋转

图 2-5 绕特定轴旋转

$${}_i^jR(X_i,\theta)=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\theta & -\sin\theta\\ 0 & \sin\theta & \cos\theta\end{bmatrix} \tag{2-11}$$

万方数据

$$ {}_{i}^{j}R(Y_i,\theta)=\begin{bmatrix}\cos\theta & 0 & \sin\theta\\ 0 & 1 & 0\\ -\sin\theta & 0 & \cos\theta\end{bmatrix} \tag{2-12}$$

$$ {}_{i}^{j}R(Z_i,\theta)=\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \tag{2-13}$$

相机绕多个坐标轴进行旋转运动时，如图 2-6。这种情况就是坐标系(X_i,Y_i,Z_i)先绕轴Z_i旋转得到坐标系(X_1,Y_1,Z_1)，旋转矩阵为$R(Z_i,\varphi)$；再绕 Y 轴旋转得到(X_2,Y_2,Z_2)，旋转矩阵为$R(Y_1,\theta)$；再绕 Z 轴旋转得到坐标系(X_j,Y_j,Z_j)，旋转矩阵为$R(Z_1,\theta)$。

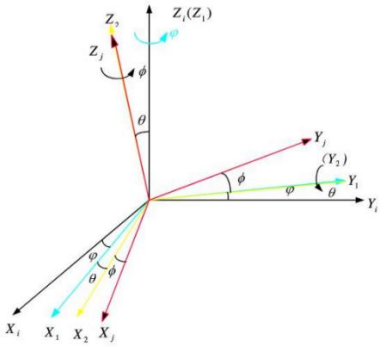


图 2-6 多轴旋转

则总的旋转矩阵，展开如下：

$$ {}_{i}^{j}R(\varphi,\theta,\delta)=\begin{bmatrix}\cos\varphi & -\sin\varphi & 0\\ \sin\varphi & \cos\varphi & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\cos\theta & 0 & \sin\theta\\ 0 & 1 & 0\\ -\sin\theta & 0 & \cos\theta\end{bmatrix}\begin{bmatrix}\cos\delta & -\sin\delta & 0\\ \sin\delta & \cos\delta & 0\\ 0 & 0 & 1\end{bmatrix} \tag{2-14}$$

3、坐标平移旋转变换

针对既有旋转又有平移的这种事件，整个运动过程通常会剥离为两种变化：

$$ P_i={}_{i}^{j}RP_j+{}_{i}^{oj}P \tag{2-15}$$

只有旋转分量时，算法上通常定义一个过渡坐标系C，令它的方向和目标坐标系i为同一个方向。于是可以将坐标系j先进行旋转变化，过渡到坐标系C，再进行一个平移分量的操作。所以参考前面的坐标变换有：

$$ P_c={}_{c}^{j}RP_j={}_{i}^{j}RP_j \tag{2-16}$$

只有平移分量时，也就是平移到坐标系j下，变化后的结果为：

$$ P_i=P_c+{}_{i}^{oc}P={}_{i}^{j}RP_j+{}_{i}^{oj}P \tag{2-17}$$

万方数据

2.2 三维视觉里程计的运动估计

三维相机通过获取的二维彩色图、深度数据图可以融合成三维点云图，如图 2-7 所示左图为二维颜色信息图，中间是三维深度信息图，右图是彩色图和深度图配准图。图 2-8 为融合后的三维点云图。

本节简要介绍投影模型和畸变的原理，从一个世界坐标系中的点 p 出发，把相机的内外参数和畸变都考虑进来，最后投影成像素坐标，

1、首先，把世界坐标转换到相机坐标，这里将用到相机外参数$\{R,t\}$：

$$P' = Rp + t = [X', Y', Z']^T \tag{2-18}$$

2、然后，将 P' 投至归一化平面，得到归一化坐标：

$$P_c = \left[u_c, v_c, 1\right] = [X'/Z', Y'/Z', 1]^T \tag{2-19}$$

3、对归一化坐标去畸变，得到去畸变后的坐标。这里暂时只考虑径向畸变：

$$\begin{cases} u_c' = u_c(1 + k_1 r_c^2 + k_2 r_c^4) \\ v_c' = v_c(1 + k_1 r_c^2 + k_2 r_c^4) \end{cases} \tag{2-20}$$

4、最后，根据内参模型，计算像素坐标：

$$\begin{cases} u_s = f_x u_c' + c_x \\ v_s = f_y v_c' + c_y \end{cases} \tag{2-21}$$

根据上述步骤可以把全局坐标系下的三维坐标点 p 进行像素坐标转换，同理也可以将像素坐标转变为世界坐标。

本文运用相机的内参和外参，可以计算出任何像素在相机坐标系下的位置。同时，根据相机外参，又能计算这些像素在世界坐标下的位置。最后通过这种关系转换就能得到真实世界的平移距离和旋转角度。

设在t时间点和$t+1$时间点捕捉到相邻场景的 RGB 图像信息、深度图像信息。本文利用相邻帧的图像特征点，以及深度信息进行运动估计，求出帧与帧之间的变换关系，得到帧与帧之间的变换关系 R、T，再利用非线性优化，回环检测等算法完成三维地图的创建与定位。

图 2-7 三维场景数据

万方数据

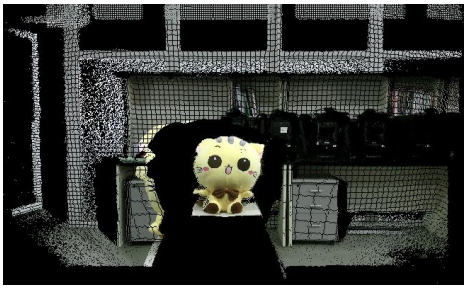


图 2-8 三维点云图

2.3 三维地图创建与定位原理

计算了视觉里程计位姿和坐标变换信息后，本文利用位姿和坐标变换关系进行三维点云的一个拼接从而进行三维地图的创建。本节主要研究三维地图创建的更新机制和定位。每一个地图都有一个自己的主方向，或者是对于世界的一个坐标系参考。所以三维地图创建之前，首先要建立主坐标系，通常把获取到的第一帧图像的相机位置，作为三维地图主坐标系。如图 2-9 所示，定义第一幅图像帧的相机位置的坐标系 $c1$ 为主坐标系，保持主坐标系和世界坐标系方向一致，其中 X 轴，Y 轴，根据任务规划可以指定横，纵向，其中 Z 轴代表视觉环境的深度方向。最后在视觉里程计运动中可以计算得到的各个状态之间的变换矩阵 R、T，再利用坐标变换，就能实现三维地图的逐帧状态解析。

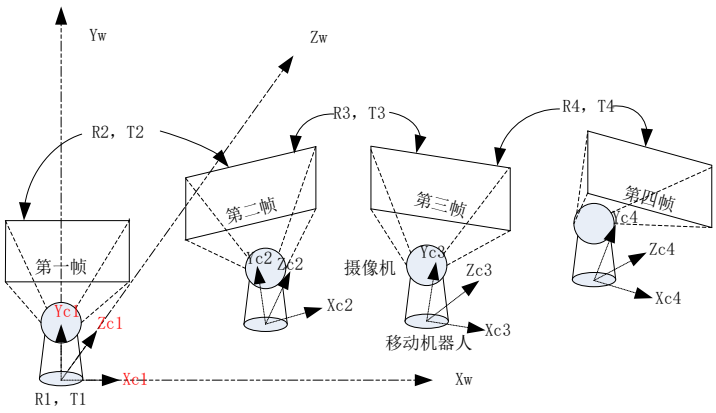


图 2-9 三维地图创建及原理示意图

2.3.1 三维地图创建原理

三维地图坐标系确立后，三维地图的创建就显得有条有理，通过视觉里程计运动状态估计可以获得每一帧图像和相邻图像之间的变换矩阵 R 和 T，同时利用图像帧和深度帧构建稠密或者稀疏点云 PointCloud，逐步获取各个帧，就能够得到一组图像之间的运动变化点云序列。最后根据点云序列和变化矩阵利用点云拼接算法

万方数据

(ICP 算法)就可以将点云序列组织成为一个三维地图。如图 2-10 所示。

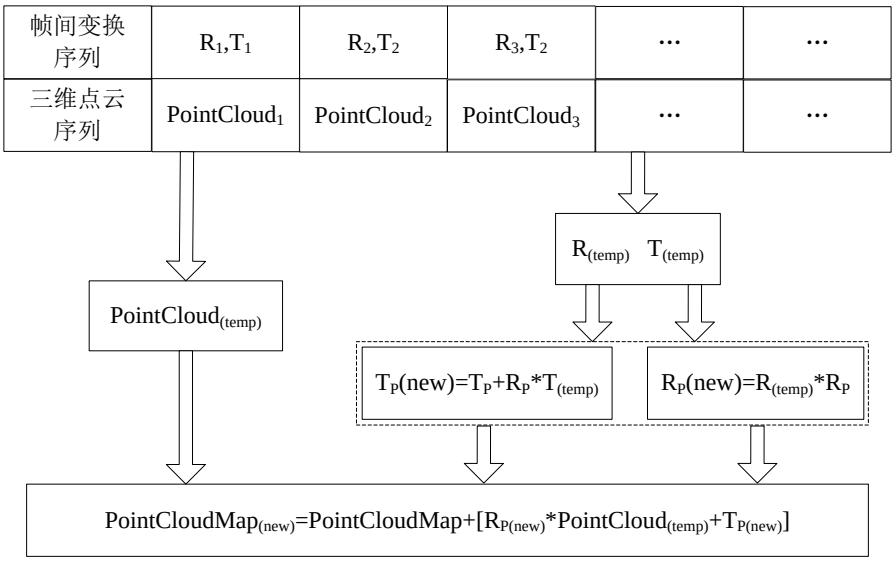


图 2-10 三维地图点云拼接流程

首先命名创建的点云地图名称为 PointCloudMap，同时定义三个存储位置为 $\mathrm{Pointcloud(temp)}$，$\mathrm{R(temp)}$，$\mathrm{T(temp)}$。存储位置主要负责接收帧间变换序列和三维点云序列的数据。接下来详细介绍前三帧三维地图创建的过程：

1、完成三维地图创建的全局坐标系设定。对应图 2-9 中的第一幅图像。首先各图像帧，旋转关系矩阵，平移关系矩阵分别输入到存储位置准备初始化，然后把 PointCloud1、R_1、T_1三个检测出的变量分别赋值给 $\mathrm{Pointcloud(temp)}$、$\mathrm{R(temp)}$、$\mathrm{T(temp)}$量，也就完成了将第一帧相机坐标系向全局坐标系的导入。

2、实现第一帧与第二帧之间的地图创建。首先将两序列中的下一个元素分别读入存储位置。如图 2-10 所示，$\mathrm{T(temp)}$与 R_p 和 T_p 是一个线性乘积累加和关系，容易得到 $T_p(new)$，同理 $\mathrm{R(temp)}$与 R_p 相乘结果为 $R_p(new)$，这其实就是一个旋转平移的过程，最后根据坐标关系的变换进行点云拼接得到地图创建结果 $\mathrm{PointCloudMap(new)}$。

3、完成将 PointCloud3 变换叠加到 PointCloudMap。将序列中的下一个元素分别读入存储区。类似于步骤二的计算关系得到新的 $R_p(new)$，$T_p(new)$值，根据新的参数值，将新的 $\mathrm{PointCloudMap(new)}$同 PointCloudMap 进行拼接，最后再次得到新地图创建结果 PointCloudMap (new)。

通过迭代上面的步骤，就可以完成三维地图的创建。

2.3.2 实时定位原理

在地图创建过程的时候，机器人需要对自己的定位进行实时更新。本文运用帧

万方数据

间变化关系，不断计算出当前位姿和当前位置，最后采用线拟合的方式快速获得机器人的一个运动轨迹。图 2-10 中虚线框部分描述了机器人位姿的更新过程，通过更新后，机器人再次获得当前姿态和位置，就能完成地图创建过程中的实时定位了。

本文用矩阵 P 和 L 来设定机器人的当前位姿的描述，其中 P 表示当前姿态，L 表示当前位置，并分别将两个变量初始化为3×3的单位矩阵和3×1的零矩阵，即以第一帧所在的位置为全局坐标系原点。将机器人每次获取帧数据时所在点的当前位置用线拟合即可得到机器人的行驶轨迹。机器人位姿的变化只与帧间变换序列相关，图 2-10 中虚线框表示了机器人位姿的更新过程，且更新后的当前姿态 P 为$R_p(new)$，当前位置 L 为与$T_p\left(new\right)$。在地图创建的过程中，同时获取$R_p(new)$与$T_p\left(new\right)$即可实现创建过程中的实时定位。

2.4 数据获取和软件开发环境

本节主要对于系统实验环境进行介绍，对于使用到的硬件和测试数据以及软件环境进行详细说明。

2.4.1 数据获取

本实验使用的是三维相机采取的数据。PC 机采用的是联想*ThinkPad E*450，其具体配置为 Intel 酷睿 i7（主频为$2.5GHz$），RAM 为 8G 内存。算法验证过程中使用了如表 2-1 所示的图像集。

表 2-1 三维场景实验数据集

数据集	TUM	Strecha	VGG	Cabiner
全名	RGB-D SLAM Dataset and Benchmark	Dense Multivew Stereo Dataset	Affine Covariant Regions Datasets	A subset of TUM dataset
图像对数	3141	500	40	578
标定好的真实数据	Camera pose，Depth	Camera pose, 3D model	Homography	Camera pose, Depth
描述	包含所有情况的图像	纹理质量高的图像	视点变化，模糊，旋转图像	强光低质感的图

2.4.2 软件开发环境

本文算法测试和验证中使用的软件环境如下：OpenCV、OpenGL、和 PCL 作为图像的基本处理工具库。其中，OpenGL (Open Graphics Library)是用于渲染 2D、

万方数据

3D 矢量图形的跨语言、OpenCV(Open Source Computer Vision Library)是一个用来做图像处理、计算机视觉以及模式识别的跨平台计算机视觉库 PCL(Point Cloud Library)是用于点云滤波、读写、存储、配准等基本操作的视觉库。

2.5 小结

本章节主要阐述了三维地图创建的相关机制，从视觉里程计的位姿和坐标变换描述到运动估计，定位与建图等几个角度进行了粗略分析，同时还介绍了三维地图的更新步骤，以及本文的一些实验条件和实验图像集。通过第二章的介绍为后续分析主要环节产生的问题，以及算法验证和改进作了一个铺垫。

万方数据

第三章 基于贝叶斯网格统计改进视觉特征匹配算法

相机是三维信息获取源，它将三维世界中的真实场景拍摄成图像，绝大多数计算机视觉研究都是以图像为研究对象的，本文所研究的三维视觉技术同样是以相机拍摄的图像作为研究对象的，基于视觉 SLAM 方向的研究主要集中在单目，双目相机或者深度相机采集的数据，无论使用什么摄像头，内在原理都脱离不了三维几何关系的计算，都必须面对图像的本真结构，都需要去消除环境干扰，从复杂环境中提取出最能代表图像的关键特征。本章主要研究图像的特征提取算法，同时设计出贝叶斯网格统计模块，模块改进了传统图像特征提取与匹配算法的效率和鲁棒性。

3.1 图像中的特征

三维视觉 SLAM 的视觉前端也就是视觉里程计。视觉里程计的作用是负责图像的采集、特征点提取，帧与帧之间的运动估计。在视觉里程计中，由于特征点容易提取和管理，所以在视觉 SLAM 系统中经常将特征点当作路标来使用。

特征点由关键点和描述子组合而成。我们把图像中具有方向，大小等信息的特征点定义为关键点，把描述关键点周围像素信息定义为描述子，一般用向量来表示。在三维视觉 SLAM 中我们需要从图像中提取性能稳定的特征点，目前主流的特征点提取主要分为基于梯度直方图的局部描述子提取和基于二进制位串的局部描述子提取。

3.1.1 基于梯度直方图的局部描述子

基于梯度直方图的局部描述子算法，通过对图像局部区域进行一个梯度计算，量化统计图像梯度为直方图，利用直方图来代表图像整体的一个梯度和幅值特性，最后将直方图中最能代表图像的部分提取出来形成特征描述向量。本小节着重阐述 SIFT 特征点描述子。

SIFT 特征点的提取步骤如下：

1、建立高斯差分尺度空间

通过实验已经证明尺度变换可以由不同的高斯卷积核来表达，如公式 3-1 图像的尺度空间为：

$$L(x,y,\sigma)=G(x,y,\ \sigma)*I(x,y) \tag{3-1}$$

万方数据

公式 3-1 中：$G(x,y,\ \sigma)$是尺度变化的高斯函数。

$$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)}/2\sigma^2 \tag{3-2}$$

公式 3-2 里(x,y)是图像的像素坐标，σ是图像的尺度坐标，用来设计不同尺度值下图像状态。

公式 3-3 是尺度空间的表达，举个形象的例子，如果在不同尺度下我们都能够根据某些稳定点来确定图像的属性，那这些点是不是就是图像的本质特征点。同时也是波动性叫小的点。所以我们构建尺度空间，就是为了去挖掘这些波动量较小的点。最后建立如图 3-1 所式的图像金字塔，利用层塔关系寻找关键图像点。

$$\begin{aligned}D(x,y,\sigma)&=(G(x,y,k\sigma)-G(x,y,\sigma))*I(x,y)\\&=L(x,y,k\sigma)-L(x,y,\sigma)\end{aligned} \tag{3-3}$$

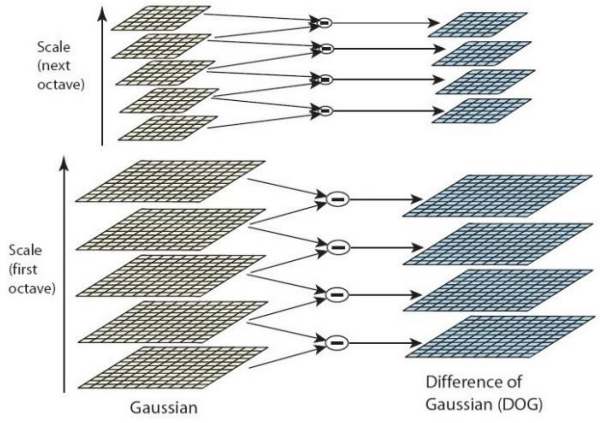


图 3-1 图像金字塔

2、尺度空间极值点检测

在DOG图像金字塔空间搭建完成后，就可以开始寻找空间波动最小的特征点，对于波动稳定点，我们很容易想到波峰或者波谷出的极值点。在尺度域上对于任一个图像像素点，我们可以在图像域及尺度域的相邻点上进行灰度值大小的判断。一般采用上下邻域9×2的相邻点，和本身 8 领域点，一共 26 个像素点进行比对。如图 3-2。

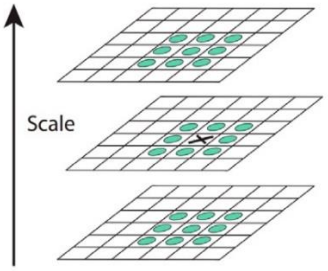


图 3-2 DOG 尺度空间局部极值点检测

万方数据

3、优化极值点位置，消除不稳定因素

选取二阶泰勒公式是我们常用的一种对噪音的剔除方法 ，为了优化特征提取点的位置，消除干扰位置对后续的算法的影响。如公式 3-4 为$D(x,y,\sigma)$的二阶泰勒拟合。

$$D(x)=D+\frac{\partial D^T}{\partial x}x+\frac{1}{2}x^T\frac{\partial^2 D}{\partial x^2}x \tag{3-4}$$

公式 3-4 中$x=(x,y,\sigma)$是从特征点的偏移量。

$$D(x')=D+\frac{\partial^2 D^{-1}}{\partial x^2}\frac{\partial D}{\partial x} \tag{3-5}$$

公式 3-5 中x'为x的导数，同时也是特征点的精确位置。

为了去除对比度较低的响应点，通过把公式 3-5 代入 3-4 得：

$$D(x')=D+\frac{1}{2}\frac{\partial D^T}{\partial x}x' \tag{3-6}$$

若$|D(x')|\geq 0.03$，则该特征点保留下来，反之，就要删掉该点。

同时为了获取稳定的边缘响应点，可以利用二阶 Hessian 矩阵边缘方向特性，边缘方向特性可以让真正的特征点在垂直边缘呈现出一个较小的变化，而在边缘方向上获得较大的一个变化。通过这种变化响应，我们可以辨别真假特征点，于是定义为如公式 3-7 的矩阵H：

$$H=\begin{bmatrix}D_{xx} & D_{xy}\\ D_{xy} & D_{yy}\end{bmatrix} \tag{3-7}$$

公式 3-7 中D_{xx}是图像上面对应的x处的二阶偏导数，D_{yy}为图像上对应的 y 方向的二阶偏导数，D_{xy}表示图像先在x方向求偏导，之后对 y 方向求偏导。Hessian 矩阵特征值其几何对应的就是像素值在x方向以及 y 方向的主曲率的变化。综上所述，我们可以直接使用H矩阵的行列式和迹来排除边缘响应点。于是得到公式 3-8 和公式 3-9，分别是H矩阵的行列式的值和迹。

$$Det(H)=D_{xx}D_{yy}-(D_{xy})^2=\alpha\beta \tag{3-8}$$

$$Tr(H)=D_{xx}+D_{yy}=\alpha+\beta \tag{3-9}$$

当两个主方向的主曲率逐步接近的过程，也就是特征点即将出现的时候。最后我们可以通过两个方向的主曲率之比r实际量化，来判断某个点是否为特征点。

万方数据

$$\frac{Tr(H)^2}{Det(H)}=\frac{(\alpha+\beta)^2}{\alpha\beta}=\frac{(r\beta+\beta)^2}{r\beta^2}=\frac{(r+1)^2}{r} \tag{3-10}$$

根据上式可以发现 r =1 时，这个比值是最小的，而且随着 r 的增大而增大。所以可以得到公式 3-11。

$$\frac{Tr(H)^2}{Det(H)}<\frac{(r+1)^2}{r} \tag{3-11}$$

根据实验总结发现，取 r =10，大于该阈值的点都是不稳定的边缘响应点，应该剔除。

SIFT 特征点的描述过程分为两步：

1、获取特征点主方向

每一个特征点都可以用特征向量表示，但是每一个向量都存在着各自的方向，对于匹配这显然是增加了运算量，所以我需要选择一个基准主方向也就是能量最大的方向，这个方向代表了图像的整体特性。公式 3-12 以及 3-13 分别计算了图像内 $L(x,y)$ 中每个特征点的方向和梯度：

$$m(x,y)=\sqrt{\left(L(x+1,y)-L(x-1,y)\right)^2+\left(L(x,y+1)-L(x,y-1)\right)^2} \tag{3-12}$$

$$\theta(x,y)=\alpha\tan 2\left(\left(L(x,y+1)-L(x,y-1)\right)/\left(L(x+1,y)-L(x-1,y)\right)\right) \tag{3-13}$$

首先设置一个边长为 1.5α，以特征点为中心的方形邻域。我们把邻域内每一个像素的幅值和方向作为我们的统计量。同时以 10° 步长，范围为 0°~360° 进行量化，得到如图 3-3，某个特征点邻域内像素点的梯度直方图统计。

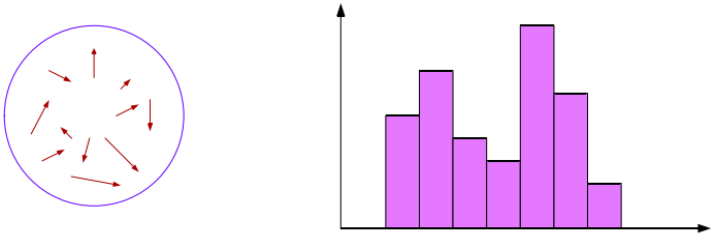


图 3-3 梯度直方图确定特征点主方向

于是我们可以定义特征点的主方向为最高峰的条块对应的角度，同时可将高于最高峰值 80%的条块作为特征点的辅助方向。最后将坐标，主方向，尺度三要素同特征点结合就成了特征描绘子。

2、特征点描述子的描绘

首先为了保持旋转不变性，我们需要建立统一的坐标体系，通常将坐标轴旋转

万方数据

为关键点的方向。然后设立8×8的网格窗口。于是关键点分布在如图 3-4 的网格中，然后我再对网格进行4×4的小网格划分，统计滑动4×4小网格内的梯度方向和幅值信息，这里对于每个网格中心我们只计算 8 个方向，所以我们在每个网格中心都获得了一个 128 维度的特征描述子。这个就是 SITF 特征点。

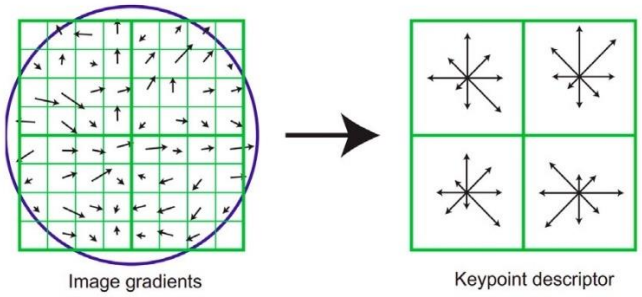


图 3-4 SIFT 特征描述子生成

3.1.2 基于二进制位串的局部描述子

基于梯度直方图方法的 SIFT 有很优秀的性能，因此它得到了非常广泛的应用。近几年，对 SIFT 描述子的各种改进算法有很多，SURF[26]，PCA-SIFT[27]，ASIFT[28]。因为应用场合的复杂性导致基于梯度直方图的特征描述子提取的计算速度和存储问题愈发严重，高性能且高速低存储的特征描述算法的需求不断上升。于是逐步有研究人员将像素以二进制表示，应用这种表达方式比较计算特征描述符直接而且具有计算量小。近两年的性能优秀的基于二进制位串的局部描述子主要以 ORB 为代表。

ORB 特征点的提取步骤如下：

1、特征点检测

ORB 特征点检测最常用的方法就是关于图像角点的检测。图像的角点是一种图像关键点的特殊表达。角点通常是利用像素点邻域灰度值间的差别性进行判别。本文使用了一种 FAST 图像角点检测算法，如图 3-5 的圆周，点 p 为圆周中心，半径为 r，圆周上一共有 n 个互相联系的像素I_k，$k=1,2,...,n$。角点的差异性通过公式 3-14 进行阈值判断。根据研究人员的测试发现 FAST 算法的运算效率会比 DOG 的运算效率高出几倍，所以 FAST 算法已经成为了主流的角点算法之一。

$$CRF=\begin{cases}1 & if\left|I_p-I_k\right|>t\\ 0 & else\end{cases} \tag{3-14}$$

其中I_p为圆周中心 p 的灰度值，I_k为圆上各个位置的灰度值，t 是角点判决阈值，通常 t 一般取 12。当 CRF=1 的统计个数超过了给定的阈值 t，此时将这个圆周点定义为候选点。

万方数据

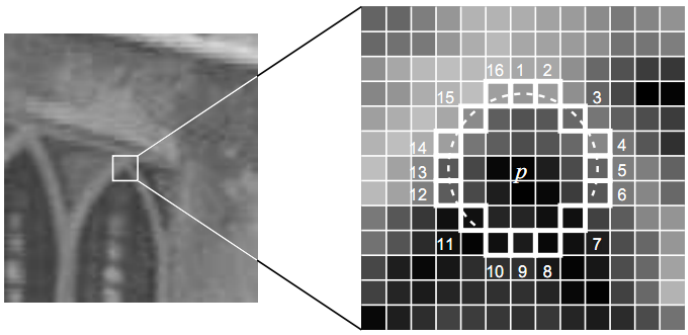


图 3-5 角点领域算子

特征兴趣点必须符合多尺度特征，但是 FAST 检测的角点信息却不具备多尺度的特性。研究人员借鉴了 SIFT 算法思路，通过建立图像金字塔以及灰度质心来解决 FAST 算法的尺度方向问题。灰度质心法也就是计算特征点矩，利用矩方向来表达特征点的方向。

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{3-15}$$

其中质心坐标 C 为：

$$C = (C_x, C_y) = (\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}) \tag{3-16}$$

其中质心方向为：

$$\theta = \arctan(\frac{C_y}{C_x}) = \arctan(\frac{m_{10}}{m_{01}}) \tag{3-17}$$

这里的θ后续是特征点主方向的一个参考。BRIE 算法利用主方向进行特征点的提取。

2、特征点的描绘

ORB 算法提取到的特征点一般使用 BRIEF 算子进行描绘。BRIFE 描述子其实就是图像块二进制表达。首先随机选取图像关键点周围的部分点构成图像块，然后对点域构成的图像块进行二值化，最后将图像块进行一个二进制的表达转换，也就是使用若干个二进制串来表示图像块，形成最后的特征点描述子。

对于图像块的二进制转换，科研人员主要是通过一个高斯核对图像的平滑滤波，然后进行一个比较判断来获得每一位 BRIEF 描述子中二进制串的值。如公式 3-18。

$$\tau(p,x,y) = \begin{cases} 1 & if \quad p(x) < p(y) \\ 0 & otherwise \end{cases} \tag{3-18}$$

万方数据

上式 3-18 中， $p(x)$ 是图像块 p 在像素点 $x=(u,v)$ 处的灰度值， $p(y)$ 为图像块 p 在像素点在 y 处的灰度值。通过图像块的转换可以获得 n 个 (x,y) 像素位置对，最终通过以上算法能够获得一个 n 位二进制串的 BRIEF 描述。

$$f_n(p)=\sum_{1\le i\le n}2^{i-1}\tau(p;x,y) \tag{3-19}$$

BRIEF 描述子的优势在于对光照变化的适应能力。在整个提取过程中，算法比较图像相应特征点的强度差异，利用差异性进行图像微分的编码。而微分特性对于突然的光照产生的灰度变化具有较强的克服能力。同时由于二进制串的性质具有旋转不变性，所以 BRIEF 描述子具有一定的抗干扰和旋转的不变特性。

当然为了进一步增强 BRIEF 描述的稳定性和高效性，在算法层面上会将高斯离散核的尺寸设置为 31×31，这里不再使用像素点，而是使用图像块。同时我们再将每一个图像块分割出 5×5 的小图像块并且统计出小图像块的像素值的累积和。通过高斯核的相关设置 BRIEF 在噪音方面显得更为稳定，尤其是 ORB 描述子能够对敏感噪音具有较强的鲁棒性。

BRIEF 描述子使用二进制来实现了方向上的旋转不变性。现在要将 BRIEF 的描述添加到 FAST 特征点上面，ORB 算法使用了邻域图像块的主方向 θ 和关键点进行配准。我们定义一个主方向函数 R_θ，和像素位置矩阵 S。然后将任意特征点像素位置上的二进制统计到矩阵里面，如 3-20。

$$S=\begin{bmatrix}x_1 & \cdots & \cdots & x_n\\ y_1 & \cdots & \cdots & y_n\end{bmatrix} \tag{3-20}$$

最后图像特征点及周围邻域图像小块的主方向 θ 的配准函数表示为 S_θ：$S_\theta=R_\theta S$。

最终得到式 3-21 的 ORB 特征点的提取函数：

$$g_n(p,\theta)=f_n(p)\,|\,(x_i,y_i)\in S_\theta \tag{3-21}$$

在整个特征点的描述环节里，ORB 算法设置了相关角度查询表，每一个角度都被量化为了 30 个 $2\pi/30$。同时 ORB 算法将特征点的主方向角度存在于查询表内，配准函数 S_θ 会提取到查询表内的相关值计算对应的特征点描述子。

3.2 传统特征匹配算法

移动机器人移动过程中，图像的特征提取可以获得图像的关键点，为了获得运动图像之间的联系，视觉 SLAM 技术里面往往需要将图像特征点进行相关匹配获得这种运动关系转换。传统的特征匹配算法分为特征匹配，信息熵率匹配，区域块

万方数据

配准，模板相应配对。因为本文使用了三维视觉进行 SLAM 技术的研究，所以本节主要介绍主流的匹配算法：局部立体匹配和全局立体匹配。

3.2.1 基于特征区域的立体匹配算法

在 3.1 节中介绍图像的两种特征描述，两种特征描述在旋转变换，噪音影响方面以及光照变化上都表现出了一定的稳定性。特征匹配算法以 SIFT 和 ORB 描述子作为匹配基元显然是合适的。在移动机器人移动过程中，图像的特征中的角或者线变化是不会发生形变的，在不失实时性的情况下，通过特征匹配算法能够稳定计算出机器人的运动关系。在双目相机中能够利用这种关系计算出运动图像的稀疏视差图。三维摄像机里面利用运动关系可以更快的进行点云的相关配准。

在传统的立体匹配算法流程上，第一步就是提取实时运动的两幅特征域图像；第二步，将当前时刻的图像为参考图像，同时计算后续运动图像的极线。第三步；当前时刻的图像极线上的每一个特征点同下一时刻的图像极线位置上的特征点进行匹配，为了获得最佳的匹配点，往往需要对匹配度进行一定的约束限定和阈值分割。最后重复上述过程，利用立体匹配算法进行运动图像极线特征点的相关匹配，得到运动关系。

3.2.2 局部立体匹配算法

局部立体匹配算法是指取一组固定大小的图像块进行匹配，而不是传统的对极线的单个特征点进行匹配。传统的单个特征点匹配过程往往存在分辨力低，同时在匹配过程中，利用单个特征点的匹配只是属于小动态范围的匹配表示，也就是说利用单个特征描述子的匹配表达还是不够清晰。在运动的过程中，局部立体匹配算法使用一组特征点像素和另一组特征点像素进行配对的结果会更加准确。对于局部立体匹配需要注意局部域上的特征点的空间问题，只有空间位置一致的特征点比较才是有意义的。

如图 3-6 所示，移动机器人获取当前图像中的任意一点 p，然后选取点 p 的邻域作为匹配窗口，然后移动机器人获取下一帧图像，并且选取同样区域的邻域，通过度量匹配邻域的距离进行匹配点的判定。

邻域搜索方面，以当前时刻图像为基准图像，在下一个时刻的图像中搜索对应的匹配。第一种邻域搜索为一维搜索，第二种为二维搜索。一维搜索就是沿着两个时刻的基线进行串口邻域搜索。二维搜索就是对两个时刻图像上根据一定规则对多个位置邻域进行匹配，同时通过匹配插值选择最好的匹配位置进行搜索。

万方数据

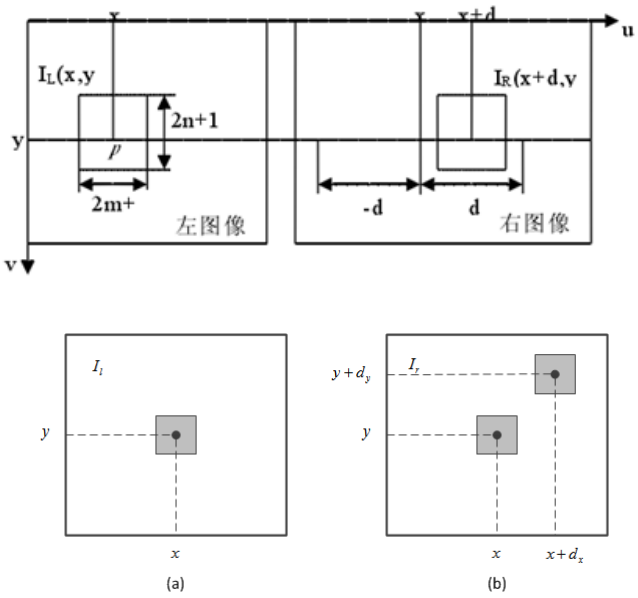


图 3-6 二维搜索的局部匹配

局部立体匹配算法常用的是 BM 匹配算法，利用图像固定图像区域块进行每一像素的匹配度度量，在匹配过程中利用度量代价函数进行局部匹配的极值求解最后确定正确的匹配位置点。具体算法过程如下：

1、对于两个时刻的图像的像素点，匹配的代价值函数为：

$$C_{AD}(x,y,d)=\left|I_L(x,y)-I_R(x-d,y)\right| \tag{3-22}$$

其中，I 为像素点的灰度值，d 为两个时刻图像匹配点对的差异值。

2、对于第一步里的代价值进行统计，具体的统计函数表达如式 3-23。

$$SAD(u,v,d)=\sum_{i=-m}^{m}\sum_{j=-n}^{n}\left|I_{L}(u+i,v+j)-I_R(u+i+d,v+j)\right| \tag{3-23}$$

3、最后求解出相似性度量函数的极值点，如式(3-21)所示。此时分别求出两个块区域的中心点即为匹配点对。

3.2.3 全局立体匹配算法

局部匹配算法仍然存在匹配的误差性，而全局立体匹配则是更为全面的一种选择。全局立体匹配通过少量的约束规则和局部的代价差值进行最佳匹配选取。在整个优化过程中，全局立体匹配算法去掉了代价统计，而是将代价值和差值能量函数进行结合，在最后的评估环节中，我们使用差值能量函数进行最后的优化判定，如式 3-24 为立体匹配的差值能量函数：

$$E(\theta)=E_{data}(\theta)+E_{smooth}(\theta) \tag{3-24}$$

万方数据

公式 3-24 中：

$$E_{data}(\theta)=\sum_{(x,y)\in I_{1,2}} S(x,y,D)，\ E_{smooth}(\theta)=\sum_{(x,y)\in I_{1,2}} \phi(x,y,|\nabla D|) \qquad (3\text{-}25)$$

公式 3-25 中，θ 为若干影响能量值的参数；$I_{1,2}$ 为待匹配的两个时刻的图像；$S(x,y,D)$ 为代价值；$E_{data}(\theta)$ 传达了参数 θ 和输入数据的差异级别；$E_{smooth}(\theta)$ 用来加强解的平滑度，该函数一般用于双目相机拍摄的视差图的额外约束，主要和光照函数有关系，而 $\phi(x,y,|\nabla D|)$ 为一个视差梯度的特定函数。

总的来说，局部立体匹配需要注意匹配窗口的尺寸和搜索方法的选取问题。窗口尺寸的大小会影响到遮挡物体的一个匹配，过大的窗口会导致匹配的正确性大大降低，过小的窗口导致采集信息量过小产生匹配上的分歧。搜索方法的选取对于计算效率是显而言之的，不同场景使用不同的搜索策略可以达到不同的匹配效率。

全局立体匹配方法在遮挡的图像表现效果更佳，利用代价差异值进行全局的判断可以获得更为准确的匹配关系。目前如置信度传播法，尺度空间法，非线性扩散等都是属于依据全局立体匹配思想来实现的。

3.3 贝叶斯网格视觉特征算法

3.3.1 贝叶斯视觉建模

对于人类视觉系统分析，假如对于模糊图像的匹配，人类视觉系统会猜测出更近似的，当然对于清晰图像人的大脑猜测速度更快，我们可以把人类视觉系统判断方式定义为一个概率问题。这个概率模型就是对于一个局部区域对于另一个区域的匹配，假如匹配的是正确区域，这种概率会更大。而匹配错误的点的概率是随机的，这个概率会比较小。

给出从同一个三维场景的不同视角拍摄的一对图像 $\{I_1,I_2\}$，对应有 $\{N,M\}$ 特征点。如图 3-7，区域 $\{I_a,I_b\}$ 为图像 $\{I_1,I_2\}$ 中两个子集区域，区域 $\{I_a,I_b\}$ 有 $\{n,m\}$ 特征点，$X=\{x_1,x_2,...x_i...,...x_n\}$ 为 I_a 到 I_b 中最邻近特征匹配子集，X 形成一个 n 维的子集。这个子集对应意味着含有 n 个像素（特征点）被识别为另一个图像中的相同点。

假设 1：平滑运动过程中物体的各个匹配领域能够找到一致的矩阵变化关系，用来建立匹配关系，但是，不匹配位置的区域的矩阵变化关系是不一样的。

由此本文将人的直观判决转为贝叶斯模型，通过观察一些正确的匹配的图像可以发现，一个小区域范围内的点匹配正确，一定会有区域内其他点的支持，当然也会存在 a 区域的点匹配错误但是仍然在相对应的 b 区域中，而真正匹配错误的

万方数据

点其错误的区域差异很大。于是可以利用概率统计把这种错误概率加大的点排除掉。

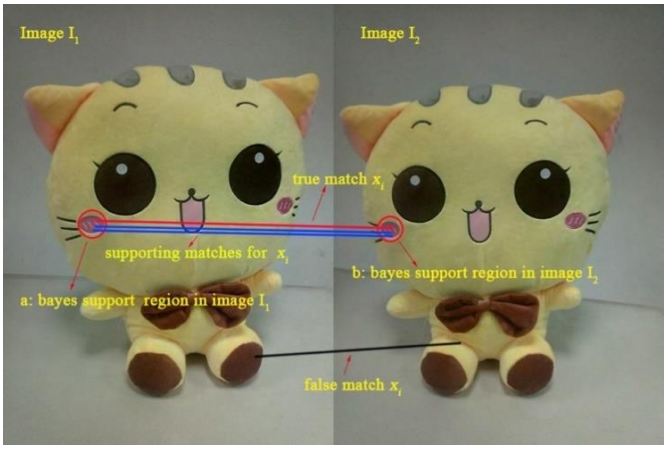


图 3-7 贝叶斯视觉建模

因此，本文定义了三种贝叶斯视觉事件：

条件一：f_a 代表区域 a 中 n 个支持特征中的一个，f_b 代表区域 b 中 n 个支持特征中的一个。

条件二：定义 f_a^b 为区域 I_a 点匹配到区域 I_b 中最邻近的对应点。

事件一：定义 f_a^t 为区域 I_a 中正确匹配到区域 I_b 中的对应点的情况，这个事件的概率为 $p(f_a^t)=t$。

事件二：定义 f_a^f 为区域 I_a 中错误匹配到区域 I_b 中的对应点情况，这个事件概率为 $p(f_a^f)=1-t$。

事件三：定义 $f_a^b \mid f_a^f$ 是错误匹配的特征点，但是仍然落在了区域 I_b 中，并且和正确对应点最邻近的情况。这个事件的概率：$p(f_a^b \mid f_a^f)=\frac{\partial n}{m}$。

同时本文定义 T_{ab} 代表正确匹配的区域{a, b}，因此可以得到如下推导：

$$\begin{aligned} p_t &= p(f_a^b \mid T_{ab}) = p(f_a^t \mid T_{ab}) + p(f_a^b, f_a^f \mid T_{ab}) \\ &= p(f_a^t \mid T_{ab}) + p(f_a^f \mid T_{ab}) p(f_a^b, f_a^f \mid T_{ab}) \\ &= p(f_a^t) + p(f_a^f) p(f_a^b \mid f_a^f) \\ &= t + (1-t)\partial n / m \end{aligned} \tag{3-26}$$

公式(3-26)第一行，因为两张图的每一个特征点是两个相互独立的，所以 $p(f_b^t)$，$p(f_b^f)$ 为两个独立的概率，所以第一行可由贝叶斯规则展开得到。又因为 $p(f_a^b \mid f_a^f)$ 一个独立的事件. 因此再次根据贝叶斯规则可得到最终的结果表达式。

类似的，本文推导出定义 T_{ab} 代表{a, b}为错误匹配的区域的概率：

万方数据

$$\begin{aligned} p_f &= p(f_a^f \mid F_{ab}) = p(f_a^b, f_a^f \mid T_{ab}) \\ &= p(f_a^f \mid T_{ab})\, p(f_a^b, f_a^f \mid T_{ab}) \\ &= p(f_a^f)\, p(f_a^b \mid f_a^f) \\ &= (1-t)\partial n / m \end{aligned} \tag{3-27}$$

因为事件 f_a^b 是事件 f_a^f 的子集。因此可以得到 $p(f_a^b \mid F_{ab}) = p(f_a^b, f_a^f \mid T_{ab})$。类似由贝叶斯规则，子集 F_{ab} 也具有独立性，转化后得到最后的表达式。

由于每个特征的匹配是独立的，利用(3-26)，(3-27)式，本文采用一对二项分布近似邻域内的匹配数目 x_i 的分布 S_i。

$$S_i \sim \begin{cases} B(n, p_t), & \text{if } x_i \text{ is true} \\ B(n, p_f), & \text{if } \mathrm{x}_i \text{ is false} \end{cases} \tag{3-28}$$

3.3.2 贝叶斯网格统计算法

首先，本文将整个图片划分为多个区域块，并且图像的每个局部区域都是独立的。许多研究表明局部图像匹配不会影响全局图像匹配。平滑运动会导致围绕真正匹配的（小）邻域计算出相同 3D 位置几何关系。同样，错误匹配周围的邻域也会看到几何不同的 3D 位置关系。如图 3-8 所示，两幅图像都用预定义的网格分割，计算网格对的运动统计量而不是每个特征的对应关系。

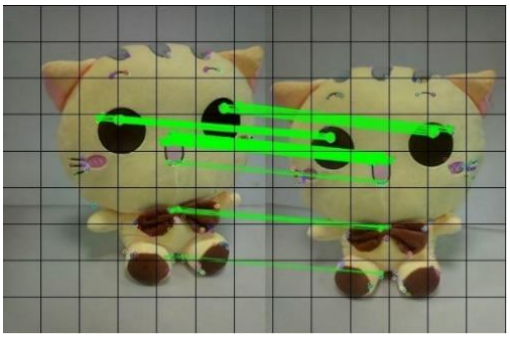


图 3-8 图像网格切分图

这里，邻域被定义为围绕如图 3-9 中所示的各个图像特征的匹配对 $\{a,b\},\{a,c\},\{a,d\}$。

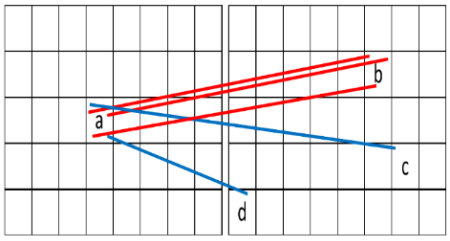


图 3-9 运动统计的单元对

万方数据

具有相同的 3D 区域，因此在两幅图像之间共享许多相似的特征，这导致邻里有许多支持匹配。相反，错误匹配邻域查看不同的 3D 区域具有更少的相似特征，这导致匹配支持的减少。

每个网格有多少匹配，那么对应的匹配网格也应该有相应数量的匹配，并且同一网格中的其他点将支持这种效果。错误点只是使得相同的错误的概率相对较低，也就是说，一个网格可能是一个错误点，但其他点对应的是相应的圆圈右边的匹配。我们定义 S_i 为同一区域内点集支持的衡量。所以，当有 k 个相匹配的不相交区域一起移动时， $x_{a^k b^k} \in x$ 是落在区域对 $\{a^k, b^k\}$ 上的匹配的子集。

所以类似的本文定义了 k 个区域，每个区域块的一个二项式分布如公式 3-29：

$$S_i \sim \begin{cases} B(kn,\ p_t), & \text{if } x_i \text{ is true} \\ B(kn,\ p_f), & \text{if } \mathrm{x}_i \text{ is false} \end{cases} \tag{3-29}$$

所以 S_i 分布的平均值和标准差为：

$$\begin{cases} m_t = knp_t,\ s_t = \sqrt{knt(1-\ p_t)} & \text{if } x_i \text{ is true} \\ m_f = knp_f,\ s_f = \sqrt{knt(1-\ p_t)} & \text{if } \mathrm{x}_i \text{ is false} \end{cases} \tag{3-30}$$

通常情况下，在处理统计事件时，考虑一个事件发生在平均值或者标准偏差处，这是不太可能发生的。所以本文采用一个得分量化来进行一个处理，如公式 3-31。

$$P = \frac{m_t - m_f}{s_t + s_f} = \frac{knp_t - knp_f}{\sqrt{knp_t(1-\ p_t)} + \sqrt{knp_f(1-\ p_f)}} \tag{3-31}$$

贝叶斯运动核是计算特征点附近中心核的支持度。正如我们通常所说的那样，运动平滑度通常不会在一个小区域中找到。我们选择一个 8 区域来计算每个特征点及其周围的特征点，如图 3-10 所示。

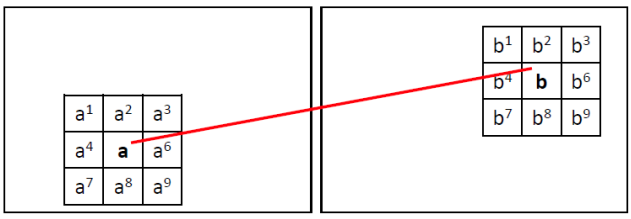


图 3-10 贝叶斯运动内核

为了解决旋转不变性，本文将带匹配的模板进行了 9 次旋转，使用九个旋转运动矩阵作为带匹配核，如图 3-11，以顺时针为方向，依次循环移动边界元素，每次移动代表 45 度的一个旋转阈值，旋转最大角度为 360 度，所以本文一共可以获得 9 个相关贝叶斯运动旋转矩阵。这样 a 矩阵可以顺利匹配到带有角度的 b 区域

万方数据

的特征点。

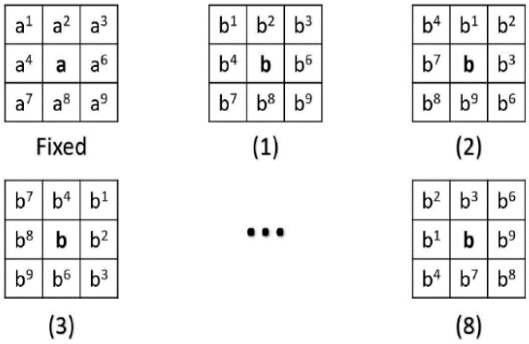


图 3-11 旋转运动核

3.4 融入贝叶斯网格统计的特征匹配算法

3.4.1 贝叶斯网格统计加速 RANSAC 算法

由于受噪声干扰等问题的影响，在特征提取过程中容易提取到干扰点。目前为止的匹配算法都会存在错误匹配点对的情况。显然误匹配点对无论是运动关系的计算还是对轨迹推算的精确度都会产生比较大的影响。因此解决错误匹配点是获得一个稳定性高的系统的关键之一。

传统的匹配筛选算法主要是 BF 和 FLANN，BF 和 FLANN 两者的区别在于 BF 会遍历所有匹配情况，也就是在匹配过程中逐一寻找最优匹配。FLANN (Fast Library for Approximate Nearest Neighbors)，是一种近似最优匹配法，也就是 FLANN 不会试图穷尽的去寻找最佳匹配，而是在时间复杂度下作了一个平衡，在一定时间内去寻找最邻近的匹配。，所以往往对匹配精度要求不高的应用环境的通常会使用 FLANN。

对于传统的匹配筛选算法中，还有一种算法受到广泛使用，也就是 RANSAC 算法，首先给定数据样本，样本中存在各种数据，一种是我们需要的正确数据，还有就是错误或者是干扰数据。RANSAC 算法要做的就是利用部分数据去假设一个正确模型，然后用模型去验证数据是否能够获得正确输出。当然正确的模型获得的正确输出显然更对，所以就通过不断迭代寻找这个符合最多数据的 RANSAC 模型。

从 RANSAC 整个算法的路径上来看，风险和收益似乎是对等。风险在于高迭代性，大量的点集进行拟合，只是为了寻找一个最优参，消耗的时间有时候是不能忍受的，当然模型一旦建立，RANSAC 算法的高稳定性又是有目共睹的。综述上来困扰 RANSAC 的点在于不确定的阈值同无上限的迭代次数的对立问题。

对基础矩阵估计是三维重建、运动估计、摄像机标定以及匹配和跟踪等研究的

万方数据

基础。首先我们要明白特征匹配时间消耗主要在哪一部分，特征匹配最关键的环节是基础矩阵或者单应矩阵的计算。基础矩阵，如果匹配对的特征点，都会具有相同的基础矩阵。若匹配错误则得不到一致的基础矩阵。单应矩阵检测的特征点确实比基础矩阵少，但是检测时间却比基础矩阵多了 10ms。

由于 RANSAC 算法的复杂性，只能减少特征提取中的特征点数，虽然解决了部分时间效率问题。但是匹配精度和鲁棒性都相应的做了部分削弱。例如 SLAM 中得到的是一个稀疏地图。过少的特征点会严重影响 SLAM 地图创建的精度。

虽然 RANSAC 算法也有减少迭代次数的一些参数选择，例如获取了 N 个实验数据，现在需要寻找建立模型的 n 个数据参数，也就是 n 个关键点，可以通过调整选取数据的次数来控制模型的迭代次数。设 w 为 N 个数据中 RANSAC 模型点的比例， z 表示 k 次选取数据后，最少有一次选到的 n 个点都为 RANSAC 模型点的概率如公式 3-32：

$$z = 1 - (1 - w^n)^k \tag{3-32}$$

其中 $1 - w^n$ 表示没有选取到模型参数点的概率，$(1 - w^n)^k$ 表示 k 次选取中没有一次命中模型参数点的概率。则定义 k 为如式 3-33 所示。

$$k = \frac{\log(1 - z)}{\log(1 - w^n)} \tag{3-33}$$

公式 3-33 中 z 一般需要大于 95%。

但是把原有模型问题变成了一个概率问题，显然是不合时宜的，为了获取好的模型机制那就会导致迭代次数 k 的不稳定。那么当添加了贝叶斯网格统计模块会是样子呢？显然添加贝叶斯网格统计模块后让特征点集中在 smooth 区域，可以加快 RANSAC 等算法的收敛速度，不至于让错误的测量，错误的假设，错误的计算等原因影响特征匹配的收敛。同时贝叶斯网格统计能够承载更多特征描绘子，类似于聚类，相同区域的特征描绘子的相互支持，可以迅速将错误信息排除掉，所以能够获得更高的鲁棒性，就算是较模糊的图像也能获得更好的匹配效果，原因就在于现在可以使用更多特征描绘子了。

作为对比本文将使用结合贝叶斯网格统计的 RANSAC 算法和没有使用贝叶斯网格统计的 RANSAC 算法。经过实验表明，将贝叶斯网格统计这种思想应用到其他的特征匹配的筛选中可以加快收敛的速度。

3.4.2 基于 ICP 的点云拼接算法

通过特征匹配我们可以获得点云之间的运动关系，利用相机内参可以计算一

万方数据

对图像点云，然后根据各张图的相机外参，把点云拼接配准，生成三维地图。

本文采用了经典的 ICP 算法来实现匹配点之间的旋转平移关系，同时拼接好点云。ICP（*Iterative Closest Point*）算法是 Besl 等[30]于 1992 年提出解决曲面配准问题的方法，后面研究人员将这种方法应用到了三维点云的最近点配准上并取得了很好的效果，ICP 算法是三维视觉 SLAM 技术中应用最多的点云配准方法，其优势在于环境的适应性好，点云拼接的效果较好。

在特征匹配过程中我们能获取到的初始移动机器人的运动关系，为了提高点云配准的精确度，需要进一步优化该变换的值。ICP 算法的目的就是通过多次迭代运算进行运动变化的优化，在每一次迭代运算过程中，对源点云的数据集上的每个点，在目标点云的数据集中寻找最近的欧式距离点，计算最近的欧式距离点后再次计算机器人运动变化关系$[R|t]$，然后将得到的运动变换应用到目标点云的数据集上，于是不断进行这个优化迭代并更新到新点云的数据集上，最终得到最优的旋转平移关系矩阵，实现连续时刻上的两个点云的精确配准。

重叠问题是点云配准中的核心环节，我们通过 ICP 算法可以获得较好的点云旋转平移关系，然后将两个点云的重合点进行拼接，为了达到较好的拼接效果，还需要进一步细化，常用的算法是单位四元数法。

1987 年 Horn[28]提出了单位四元数法，算法的核心思想为利用两个重合点集上的相同点以及相关关系去寻找最高重合的点云集合。单位四元数算法的主要步骤为：旋转向量$\mathrm{q}_R=[q_0 q_1 q_2 q_3]^T$采用了单位四元数来表示，其中$q_0>=0$且$q_0{}^2+q_1{}^2+q_2{}^2+q_3{}^2=1$，此时旋转矩阵变为了一个$3\times3$的旋转矩阵$R(\mathrm{q}_R)$。在单位四元数法中假定平移矩阵为$q_T=[q_4 q_5 q_6]$，那么刚体变换的相关矩阵就变成$q=[\mathrm{q}_R|q_T]$。最后为了让两个点云集合点中对应点之间的距离最小，单位四元数法使用了公式 3-34 来使刚体变换的求解最小化。

$$f(q)=\frac{1}{N_p}\sum_{i=1}^{N_p}x_i-R(\mathrm{q}_R)p_i-q_T \tag{3-34}$$

其中矩阵$R(\mathrm{q}_R)$为：

$$R(\mathrm{q}_R)=\begin{bmatrix} q_0{}^2+q_1{}^2-q_2{}^2-q_3{}^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0{}^2-q_1{}^2+q_2{}^2-q_3{}^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3-q_0q_1) & q_0{}^2-q_1{}^2-q_2{}^2+q_3{}^2 \end{bmatrix} \tag{3-35}$$

计算出矩阵$R(\mathrm{q}_R)$后，对目标点云的数据集进行该矩阵变换使得两个点云的数据集获得最大的重合度。

综上所述，ICP 算法的步骤可归纳为：

万方数据

（1）获取待配准点云上的相关点的数据集 Plk，然后寻找目标点云上最近的点云的数据集 Prk。

（2）对拼接的两个点云的数据集进行中心化处理，得出点云的数据的相关重心位置。

（3）计算两个中心处理后的点云的数据集的正定矩阵 N 和最大特征值，以及对应的特征向量。

（4）转换四元数为旋转矩阵，并计算得到最小的单位四元数和四元数旋转矩阵中最大的特征向量。

（5）确定两个点云之间的重心点坐标和旋转矩阵。

（6）迭代结束的数值判定。利用待配准的两个数据集进行欧式距离平方和平方差值的累加 ω 进行结束阈值的判定。

（7）如果 ω 小于预先给定的阈值，则停止迭代，否则，重复步骤（1）到步骤（6），直至满足条件后停止迭代。

本文从 TUM 数据集中任意选取了如图 3-12 的两幅不同方位的彩色图像，通过深度数据，利用 ICP 算法完成了两幅图像在三维空间的一个点云拼接，如图 3-13。

图 3-12 不同方位拍摄下来的彩色数据图像

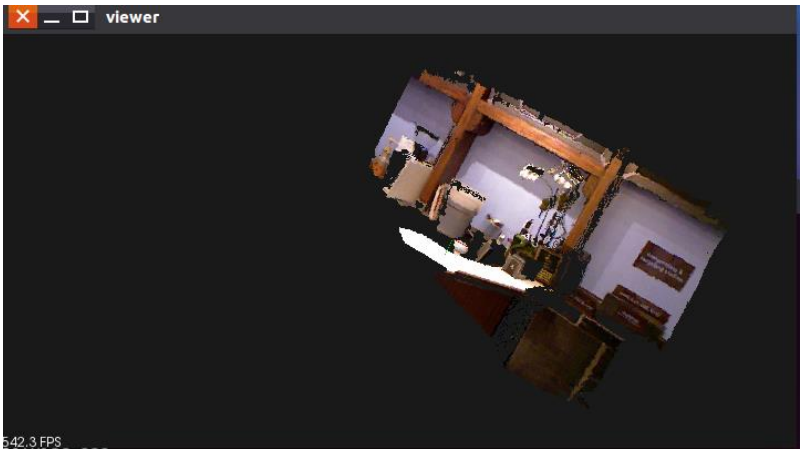


图 3-13 基于 ICP 算法的点云拼接实验结果

万方数据

3.5 实验结果与分析

根据上述贝叶斯网格统计匹配的算法思想，本文绘制出如图 3-14 的一个算法流程图:

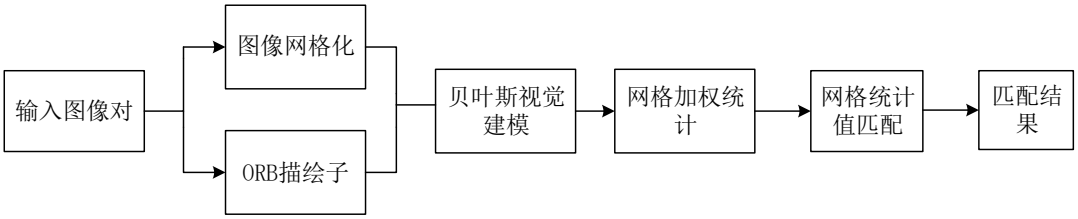


图 3-14 贝叶斯网格统计匹配算法流程

这里本文简单利用自己采集的数据验证整个贝叶斯网格统计视觉匹配算法的合理性，更多的实验数据集对比，统一放在了第六章。

图 3-15 分别为三组 RGB 图像进行特征提取与匹配的测试，左中右分别为 SIFT，ORB，贝叶斯视觉特征提取和匹配(BGS)的实验结果，其相关实验数据见表 3-1。

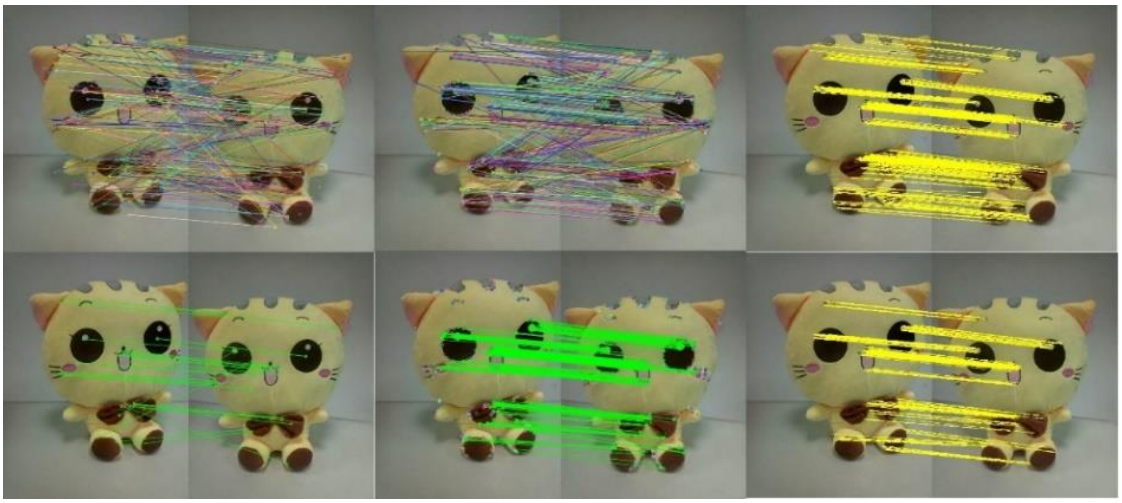


图 3-15 图像特征提取与匹配实验结果图

表 3-1 特征提取与匹配实验结果对比

特征点提取与匹配方法	特征点数	特征点提取时间(ms)	匹配对的数目	特征点匹配时间(ms)	总时间(ms)
SIFT+RANSAC	500	83.91	168	10	225
ORB+RANSAC	1112	14.92	413	15	32
BGS+RANSAC	1112	14.92	512	1.5	27

通过表格数据可以发现本文的算法初步具有执行性和合理性。在第六章，大量

万方数据

数据集的测试结果会进一步验证贝叶斯网格统计融入匹配筛选的可行性，在特征提取上使用了 ORB 的特征点，进行贝叶斯网格统计匹配，过程大大加速了匹配过程，由于网格类似并行化运算，让整个匹配总时间消耗也达到了优化。从实验中可以看出利用网格统计的特性却大大加快了错误点的一个剔除过程。证实了我们算法的一个可行性。

实际上，在实验过程中本文利用大量的数据集进行测试，发现更多的网格单元可以提高匹配本地化。然而，它减少了 n，即每个单元中的特征数量，从而减少了分割能力。这可以通过增加 k 来补偿，但会影响计算时间。我们的理论和实证结果表明 10000 个特征的 G = 20×20 个细胞。这给出 n 的平均值为 25。更多特征允许更精细的单元。根据网格数量的限制，我们基于实验获得了相关的阈值获取方法。

本文定义阈值 S_{ij} 可以被用来分割匹配对中的真集和假集 $\{T,F\}$。如公式 3-36，期望的阈值可以给出 $\delta = m_f + \partial s_j$，其中 m_f 是平均值，并且 ∂s_j 是适当数量的标准偏差以确保大多数错误的网格单元被拒绝。在实践中，m_f 它的设计很小，而 ∂ 非常大，可以确保大量错误的匹配对的可靠排斥。阈值可以近似为 $\delta = \delta \bullet S_f \approx \partial\sqrt{n}$。

$$\{i,\ \mathrm{j}\}\ \hat{\mathrm{I}}\ \begin{cases} T,\ \text{if } \mathrm{S}_{ij} > d_i = \P\sqrt{n_i} \\ F,\ \text{otherwise} \end{cases} \tag{3-36}$$

在测试时算法中具有 RANSAC 算法的贝叶斯网格统计(BGS)远高于其他快速解决方案，并且与 BF 和 FLANN 等解决方案相当。根据实验的经验，几乎所有纯算法的技术都可以通过 GPU 使用达到实时。但是工业机器人搭配 GPU 显然是不合适的。本文的视觉匹配算法在 CPU 中的计算时间为 1.5ms，贝叶斯网格统计保持并提升了传统 SLAM 视觉匹配上的实时性能。

3.6 本章小结

本章主要对传统特征匹配算子 SIFT 和 ORB 进行了总结和研究。通过实验发现在三维视觉 SLAM 算法中，对于经典的 SIFT 特征提取算法是无法使用的，原因在于 SITF 的特征点的的特征向量维数导致高计算复杂度，严重制约了系统的实时性。而实验结果表示 ORB 特征提取的方法在视觉里程计的实时性上面表现更好。但是 ORB 也存在一些问题，虽然 ORB 算子的描述能将特征点向量维数降低了一半，通过减少特征点数可以达到实时的使用效果。但是却丢失了部分精度，在质量差，纹理低的场景中，效果较差。如果增加 ORB 的特征点数，又会导致后序匹配筛选计算运动关系收敛速度大大降低。

对于上面的问题，通过研究 SLAM 的视觉特征和匹配过程中出现复杂度高的

万方数据

算法环节，在传统的视觉提取基础上，通过建立贝叶斯视觉模型，利用网格区域的加权统计，设计出一种贝叶斯网格统计算法模块，实验证明了该模块可以加速视觉特征匹配效率，提高传统匹配算法的鲁棒性。

万方数据

第四章 光速平差法改进非线性 SLAM 状态估计

机器人在运动过程中通过视觉里程计获取的数据只是短暂的，同时希望整个运动轨迹在较长时间内都能保持最优的状态。所以机器人需要不断去更新状态信息，告诉自己的位置。目前整个状态估计的更新优化的方法主要采用卡尔曼滤波算法。但是卡尔曼滤波没有参考之前的不同时间的状态估计信息，对于位姿状态估计的问题，采用卡尔曼滤波器算法[32]极容易导致状态估计的局部性误差问题。所以本章主要讨论光速平差法对非线性状态估计的优化。

4.1 状态估计模型

在第二章我们介绍了机器人的运动是可以采用位置和姿态来进行指代。在每一个时刻当中，机器人所搭载的视觉传感器都能够将测得的特征点构成路标信息。所以假设机器人在$t=0$到$t=N$的时间内运动，每一个时刻对应有x_0到x_N那么多个位姿，这N个位姿形成了机器人的轨迹。在建图方面，可以用多个特征点完成路标描述，构造出稀疏地图。假设路标点一共有M个，表示为$y_1,...y_M$，对于路标和位姿的变化，可以用运动方程和观测方程来描述 SLAM 过程，如公式 4-1：

$$\begin{cases} x_k = f(x_{k-1}, w_k) + w_k \\ z_{k,j} = h(y_j, x_k) + v_{k,j} \end{cases} \quad k = 1,.....N,\ j = 1,.....M \tag{4-1}$$

其中w_k为测量误差，$v_{k,j}$为像素误差。

对于公式 4-1，我们需要注意在观测方程中，只有当x_k看到了y_j时，才会产生观测数据，否则就没有。在机器人运动过程中，在一个区域通常只能看到一小部分路标，如图 4-1 所示。由于视觉 SLAM 特征点数量众多，所以实际情况观测方程数量会远远大于运动方程的数量。

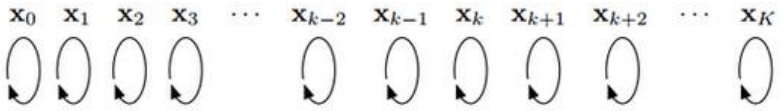


图 4-1 每一个时刻的路标观测

如果没有测量运动装置，那么也就没有运动方程。我们通过假设估计将整个优化问题变成只和若干个观测方程有关的形式。这种处理方法和 SFM[33]问题类似，本文通过使用一组图像来恢复运动和结构，区别在于，SFM 采集的图像组可以是毫无相关的，但是 SLAM 的图像序列是有时间先后的。

万方数据

4.1.1 状态估计概率模型

三维相机运动过程中会受到噪声影响导致每个方程也会存在问题，那么怎么转换这个问题？首先如果把位姿 x 和路标 y 看成服从某种概率分布的随机变量，而不是单独的一个数。那这个问题就转换为当某些运动数据 u 和观测数据 z 时，如何来确定状态量 x, y 分布？每当得到新来时刻的数据之后，它们的状态分布会变化吗？如果变化，那么需要即时更新。在比较合理的情况下，本文假设状态量和噪声项服从高斯分布，最后原问题就变成一个更新和存储均值和协方差问题。

在状态估计问题中，通常把状态估计转换为最小二乘的做法。令 x_k 为 k 时刻的所有未知量，即待估计的位姿和路标点。x_k 包含了当前时刻的相机位姿以及 m 个路标点。所以当前时刻的位姿和路标可以表示为：

$$x_k = \{x_1, y_1, ..., y_m\} \tag{4-2}$$

同时，把 k 时刻的所有观测记作 z_k。于是，运动方程与观测方程的形式可以写为：

$$\begin{cases} x_k = f(x_{k-1}, u_k) + w_k \\ z_k = h(x_k) + v_k \end{cases} \quad k = 1, ...N \tag{4-3}$$

对于第 k 时刻的情况，本文利用了 0 到 k 时刻时的数据，来估计机器人第 k 时刻的状态分布。已知初始状态 x_0，输入数据 $u_{1:k}$，观测数据 $z_{1:k}$，那么计算初始状态的条件概率如公式 4-4 所示：

$$P(x_k \mid x_0, u_{1:k}, z_{1:k}) \tag{4-4}$$

其中下标 $0:k$ 表示从 0 时刻到 k 时刻的所有数据。

按照 Bayes 法则，把 z_k 与 x_k 交换位置，得到：

$$P(x_k \mid x_0, u_{1:k}, z_{1:k}) \propto P(z_k \mid x_k) P(x_k \mid x_0, u_{1:k}, z_{1:k-1}) \tag{4-5}$$

公式 4-5 第一项为似然估计，第二项为先验估计。当前状态 x_k 是基于过去所有的状态估计得来的。所以当前状态必定会收到前一个状态 x_{k-1} 的影响。最后可以得到按照 x_{k-1} 时刻条件展开的概率公式：

$$P(x_k \mid x_0, u_{1:k}, z_{1:k-1}) = \int P(x_k \mid x_{k-1}, x_0, u_{1:k}, z_{1:k-1}) P(x_{k-1} \mid x_0, u_{1:k}, z_{1:k-1}) dx_{k-1} \tag{4-6}$$

4.1.2 基于马尔科夫假设的线性系统

根据上面的贝叶斯估计，我们还不能实际去操作它，需要将机器人运动过程简化。通过采用一阶马尔科夫假设，即机器人运动过程中 k 时刻状态只与 $k-1$ 时刻状

万方数据

态有关，而与再之前的状态无关，如图 4-2，通过这个假设我们就能从某时刻的状态估计，推导出下一个时刻。

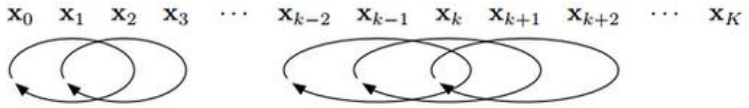


图 4-2 马尔科夫假设观测

应用马尔科夫性，也就是，公式 4-6 中等式右侧第一部分可进一步简化为：

$$P(x_k \mid x_{k-1}, x_0, u_{1:k}, z_{1:k-1}) = P(x_k \mid x_{k-1}, u_k) \tag{4-7}$$

因为k时刻状态只与$k-1$时刻之前的状态无关，所以公式 4-6 中等式右侧第二部分可简化为：

$$P(x_{k-1} \mid x_0, u_{1:k}, z_{1:k-1}) = P(x_{k-1} \mid x_0, u_{1:k-1}, z_{1:k-1}) \tag{4-8}$$

这里考虑到k时刻的输入量u_k与$k-1$时刻的状态无关，所以u_k这个状态变量可以省去。我们可以发现k时刻最后只与$k-1$时刻的状态分布有关，在程序执行的时候，我们只需要维护一个状态量，同时不断去迭代和更新这个状态量就行了。当然，如果这个状态量服从高斯分布，那么我们只需考虑迭代和更新状态量的均值和协方差即可。

最后我们可以由线性方程组来描述线性高斯系统的运动方程和观测方程：

$$\begin{cases} x_k = A_k x_{k-1} + u_k + w_k \\ z_k = C_k x_k + v_k \end{cases} \quad k = 1, \ldots, N \tag{4-9}$$

其中若假设所有的状态和噪声均满足高斯分布。记这里的噪声服从零均值高斯分布：

$$w_k \sim N(0, R), \quad \mathrm{v}_k \sim N(0, Q) \tag{4-10}$$

为了简约我省略了 R 和 Q 的下标。现在，利用马尔可夫性，假设我们知道了$k-1$时刻的后验状态估计：x_{k-1}和它的协方差P_{k-1}，现在要根据k时刻的输入和观测数据，确定x_k的后验分布。为区分推导中的先验和后验，我们记：以$\hat{x}$表示后验，以横线x_k表示先验分布。

通过运动方程确定x_k的先验分布。根据高斯分布的性质，显然有：

$$P(z_k \mid x_k) = N(C_k x_k, Q) \tag{4-11}$$

这一步称为预测，它显示了如何从上一个时刻的状态，根据输入信息（有噪声），推断当前时刻的状态分布。这个分布也就是先验。则可以写为：

万方数据

$$\hat{x}_k = A_k \hat{x}_{k-1} + u_k,\ \ \overline{\mathrm{P}}_k = A_k \hat{P}_{k-1} A_k^T + R \tag{4-12}$$

由观测方程，我们可以计算在某个状态下的观测方程如公式 4-13：

$$P(z_k \mid x_k) = N(C_k x_k, Q) \tag{4-13}$$

为了得到后验概率，我们需要计算它们的乘积，由贝叶斯公式可以得到一个关于 x_k 的高斯分布，所以我们可以设 $x_k \sim N(\hat{x}_k, \hat{P}_k)$，那么：

$$N(\hat{x}_k, \hat{P}_k) = N(C_k x_k, Q) \bullet N(\overline{x}_k, \overline{P}_k) \tag{4-14}$$

由于公式 4-14 等式两侧都是高斯分布，那么因子部分可以省去，只比较指数部分即可，指数部分很像是一个二次型的配方，所以就有：

$$(x_k - \hat{x}_k)^T \hat{P}_k^{-1} (x_k - \hat{x}_k) = (z_k - C_k x_k)^T Q^{-1} (z_k - C_k x_k) + (x_k - \hat{x}_k)^T \overline{P}_k^{-1} (x_k - \overline{x}_k) \tag{4-15}$$

将等式两边展开，比较 x_k 的二次和一次系数，对于二次系数就可以得到：

$$\hat{P}_k^{-1} = C_k^T Q^{-1} C_k + \overline{P}_k^{-1} \tag{4-16}$$

该式给出了协方差的计算过程，为了计算方便，我们定义一个中间变量：

$$K = \hat{P}_k C_k^T Q^{-1} \tag{4-17}$$

根据 4-16 左右各乘 $\hat{P}_k$

$$I = \hat{P}_k C_k^T Q^{-1} C_k + \hat{P}_k \overline{P}_k^{-1} \tag{4-18}$$

于是得到：

$$\hat{P} = (I - K C_k) \overline{P}_k \tag{4-19}$$

再比较一次项的系数，得到：

$$-2\hat{x}_k^T \hat{P}_k^{-1} x_k = -2 z_k^T Q^{-1} C_k x_k - 2\overline{x}_k^T \overline{P}_k^{-1} x_k \tag{4-20}$$

化简整理得到：

$$\hat{P}_k^{-1} x_k = C_k^T Q^{-1} z_k + \hat{P}_k \overline{P}_k^{-1} \overline{x}_k \tag{4-21}$$

两侧乘以 $\hat{P}_k$ 并带入式子得：

$$\begin{aligned} \hat{x}_k &= \hat{P}_k C_k^T Q^{-1} z_k + \hat{P}_k \overline{P}_k^{-1} \overline{x}_k \\ &= K z_k + (I - K C_k) \hat{x}_k = \overline{x}_k + K(z_k - C_k \overline{x}_k) \end{aligned} \tag{4-22}$$

于是本文得到了后验均值的表达。总而言之，上面的两个步骤可以归纳为“预测”和 “更新”两个步骤：

预测阶段：计算当前时刻的状态分布

万方数据

$$\hat{x}_k = A_k \hat{x}_{k-1} + u_k,\ \ \bar{P}_k = A_k \hat{P}_{k-1} A_k^T + R \tag{4-23}$$

更新阶段：先计算 K，它又称为卡尔曼增益：

$$K = \bar{P}_k C_k^T (C_k \bar{P}_k C_k^T + Q)^{-1} \tag{4-24}$$

然后计算后验概率的分布，得到利用最大后验概率估计优化线性系统的分布：

$$\hat{x}_k = \bar{x}_k + K(z_k - C_k \bar{x}_k) \tag{4-25}$$

$$\hat{P}_k = (I - KC_k)\bar{P}_k \tag{4-26}$$

4.1.3 非线性系统的优化处理

三维视觉 SLAM 中的相机模型下，当我们使用相机内参模型以及坐标状态方程表示的位姿时，SLAM 中的运动方程和观测方程通常是非线性函数，同时一个高斯分布，经过非线性变换后，往往不再是高斯分布。所以在非线性系统中，我们必须将一个非高斯的分布转换成成一个高斯分布。也就是把线性系统中的概率统计拓展到非线性系统中来。通常我们可以对非线性方程使用一阶泰勒展开，保留主要成分，这样就能获得线性相关的运动方程和观测方程。然后按照线性系统进行推导。令 k−1 时刻的均值与协方差矩阵为 $\hat{x}_{k-1}$，$\hat{P}_{k-1}$。在 k 时刻，我们把运动方程和观测方程，在 $\hat{x}_{k-1}$，$\hat{P}_{k-1}$ 处进行一阶泰勒展开，有：

$$x_k \approx f(\hat{x}_{k-1}, u_k) + \left.\frac{\partial f}{\partial x_{k-1}}\right|_{\hat{x}_{k-1}} (x_{k-1} - \hat{x}_{k-1}) + w_k \tag{4-27}$$

记这里的偏导数为：

$$F = \left.\frac{\partial f}{\partial x_{k-1}}\right|_{\hat{x}_{k-1}} \tag{4-28}$$

同样的，对于观测方程，有：

$$z_k \approx h(\bar{x}_k) + \left.\frac{\partial h}{\partial x_k}\right|_{\bar{x}_k} (x_{k-1} - \hat{x}_k) + n_k \tag{4-29}$$

记这里的偏导数为：

$$H = \left.\frac{\partial h}{\partial x_k}\right|_{\bar{x}_k} \tag{4-30}$$

所以预测步骤中，根据运动方程有：

$$P(x_k \mid x_0, u_{1:k}, z_{0:k-1}) = N(f(\hat{x}_{k-1}, u_k), F\hat{P}_{k-1}F^T + R_k) \tag{4-31}$$

万方数据

记这里先验和协方差的均值为：

$$\bar{x}_k = f(\hat{x}_{k-1}, u_k), \quad \bar{P}_k = F\hat{P}_{k-1}F^T + R_k \tag{4-32}$$

然后，考虑在观测中，我们有：

$$P(z_k \mid x_k) = N((h(\bar{x}_k) + H(x_k - \bar{x}_k), Q_k) \tag{4-33}$$

最后，根据贝叶斯展开式，可以推导出 x_k 的后验概率形式。通过卡尔曼滤波器原理，我们先定义一个卡尔曼增益 K_k

$$K_k = \bar{P}_k H^T (H\bar{P}_k H^T + Q_k)^{-1} \tag{4-34}$$

在卡尔曼增益的基础上，后验概率的形式为：

$$\hat{x}_{k-1} = \bar{x}_k + K_k(z_k - h(\bar{x}_k)), \quad \hat{P}_k = (I - K_k H)\bar{P}_k \tag{4-35}$$

所以我们得到了在 SLAM 非线性的情况下，单次线性近似下最大后验估计。

4.2 基于光速平差法改进非线性状态估计

4.2.1 EFK 存在的问题

在 SLAM 算法中使用的状态估计优化滤波方法目前都是基于扩展的卡尔曼滤波器(EFK，Extended Kalman Filter)[34]，其核心也是通过假设运动过程的马尔可夫性，也就是 k 时刻状态只与 $k-1$时刻相关，而与 $k-1$时刻之前的状态和观测都无关。这有点像是在视觉里程计中，只思考相邻两帧关系一样。假设当前帧的确与很久之前的数据有关，若出现这种情况，滤波器就会难以处理。而非线性优化方法则倾向于使用所有的历史数据。它不光考虑邻近时刻的特征点与轨迹关系，更会把考虑很久之前的状态也考虑进来，称为全体时间上的 SLAM[35]。如图 4-3，在这种意义下，非线性优化方法使用了更多信息，当然也需要更多的计算。

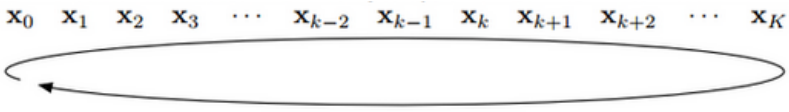


图 4-3 全局考虑所有时间点的数据

EKF 滤波器在 $\hat{x}_{k-1}$ 处取得是泰勒展开式的第一项，一阶泰勒展开只能用来描绘较近工作点的非线性情况。对于较远工作点，虽然此时后验概率依旧有效，但是 EFK 滤波器这种小范围线性近似方法，无法适用于很远的地方，极容易产生 EFK 非线性误差。所以针对 EFK 的非线性误差问题，在优化时做一阶或二阶梯度下降的近似时，需要重新对新的估计点做泰勒展开，而不像 EKF 那样只在固定点上做

万方数据

一次泰勒展开。这种优化方法适用范围更广，状态变化较大时也能使用。

EFK 还存在另一个问题，那就是计算存储问题。视觉 SLAM 中路标数量很多，每一个路标对应一组状态量的均值和方差，同时还要对它们进行维护和更新。这个存储量是巨大的，并且和状态量呈平方增长。因此，EKF 对于大型场景 SLAM 有很大的局限性。

4.2.2 光速平差法的代价函数和求解

光速平差法是在视觉恢复重建过程中，通过获取最优的三维模型和相机参数，之后利用几束穿过特征点并反射出来的光线，把相机姿态和特征点在空间位置中调整出最优的结果。最后将光束重新回撤到相机光心位置的过程。

从上看出光速平差法的求解思路完美匹配只有观测方程的 SLAM 问题。光速平差算法不光拥有较高程度的精确度，同时还具有很好的实时性效果，能够实时计算的 SLAM 场景中充分应用。但是光速平差法会使用巨量的特征点以及相机位姿。并且因为第三章的贝叶斯视觉特征提取，会导致计算量更大，所以在场景之中本文对光速平差算法进行稀疏化，让其能在实时的场景中使用。

假如现在观测像素坐标 $z=[u_s,v_s]^T$，本文定义一个观测误差函数如公式 4-36，目的是让这个误差最小，这样就能得到最佳的状态位姿估计。

$$e=z-h(\xi,p) \tag{4-36}$$

其中 x 指此时相机的位姿函数用 ξ 表示，即外参 R,t 关系。三维点 p 代表路标y。

同理把其他时刻的观测量添加进来，设 $z_{i,j}$ 为在位姿 ξ_i 处观察 p_j 产生的数据，那么整体的代价函数为：

$$\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\left\|e_{i,j}\right\|^2=\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\left\|z_{i,j}-h(\xi_i,p_j)\right\|^2 \tag{4-37}$$

公式 4-37 就是光速平差法的代价函数。

对公式 4-37 求解的过程其实也就是对路标和位姿动态调整的过程。我们可以发现函数 $h(\xi_i,p_j)$ 不是线性函数。这个时候就需要使用非线性优化手段来优化它。根据非线性优化中的梯度下降思想，首先选取某个的初值开始优化，然后不断地寻找该值的一个下降方向 Δx，从而找到目标函数的最优解，也就是不断地求解增量方程中的增量 Δx。所以在整体光速平差法目标函数上，本文将自变量定义成所有待优化的变量，而不再只是针对单个位姿和路标点，如公式 4-38：

$$x=[\xi_1,\xi_2,...,\xi_m,p_1,p_2,...,p_n]^T \tag{4-38}$$

万方数据

进一步，进行相应表达替换，增量方程中Δx 是对整体自变量的增量。所以给自变量一个增量时，目标函数变为：

$$\frac{1}{2}\left\| f(x+\Delta x)\right\|^2 \approx \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\left\| e_{i,j}+F_{ij}\Delta\xi_i+E_{ij}\Delta p_j\right\|^2 \tag{4-39}$$

公式 4-39 中 $F_{i,j}$ 表示当前状态下整个代价函数在对相机姿态的偏导数，$\mathrm{E}_{i,j}$ 代表代价函数对路标点位置的偏导。

进一步化简公式 4-39，现在将相机位姿变量整理到一个变量中如公式 4-40：

$$x_c=[\xi_1,\xi_2,...,\xi_m]^T\in R^{6m} \tag{4-40}$$

同时把空间点的变量也整理在一起：

$$x_p=[p_1,p_2,...,p_n]^T\in R^{3n} \tag{4-41}$$

于是公 4-41 可以简化表达为如下：

$$\frac{1}{2}\left\| f(x+\Delta x)\right\|^2=\frac{1}{2}\left\| e+F\Delta x_c+E\Delta x_p\right\|^2 \tag{4-42}$$

无论使用高斯牛顿法G-N [36]或者拟牛顿法L-M [37]方法，最后都将面对增量线性方程，也就是公式让 4-42 从一个由很多个小型二次项之和，变成了一个更整体的样子。于是我们将每个误差项的导数 $F_{i,j}$ 和 $E_{i,j}$ 组合起来。首先定义增量线性方程：

$$H\Delta x=g \tag{4-43}$$

本文把变量归类成了位姿和空间点两种，所以位姿和空间点构成的矩阵可以分块为：

$$J=[F\ E] \tag{4-44}$$

这里我们采用G-N 方法，则 H 矩阵为：

$$H=J^TJ\begin{bmatrix}F^TF & F^TE\\ E^TF & E^TE\end{bmatrix} \tag{4-55}$$

当然在L-M 也行，两者在算法属性上一致，只是表达方式不同。由于本文考虑了所有的优化变量，导致整个线性方程的维度非常大，包含了所有的相机位姿和路标点。如果直接应用到视觉 SLAM 中，特别是在第三章提出了贝叶斯网格统计的算法，让一个图像提取的特征点超出了传统算法的特征点数，这无疑更增加了这个线性方程的规模。如果直接对 H 求逆来计算增量方程，由于矩阵求逆是复杂度为 $O(n^3)$ 的操作，这是非常消耗计算资源的[38]。于是结合自己在第三章的变动，接下

万方数据

来我们还需要利用 H 矩阵的特殊结构的加速求解过程。

4.2.3 稀疏性和边缘化

21 世纪视觉 SLAM 的一个重要进展是认识到了矩阵 H 的稀疏结构，并发现该结构可以自然、显式地用图优化来表示[39][40]。本节将研究矩阵 H 矩阵的稀疏性对于非线性方程维度数的优化。通过观察代价函数 e_{ij}，发现该误差项只描述了在 ξ_i 看到 P_j，只涉及第 i 个相机位姿和第 j 个路标点，对其余部分的变量的导数都为 0。所以该误差项对应的有下面的形式：

$$J_{i,j}(x)=(0_{2\times6},...0_{2\times6},\frac{\partial e_{i,j}}{\partial \xi_i},0_{2\times6},...0_{2\times3},\frac{\partial e_{i,j}}{\partial p_j},0_{2\times3},...0_{2\times3}) \tag{4-46}$$

如图 4-4 两个位姿和六个路标点，则此时 J 和 H 为：

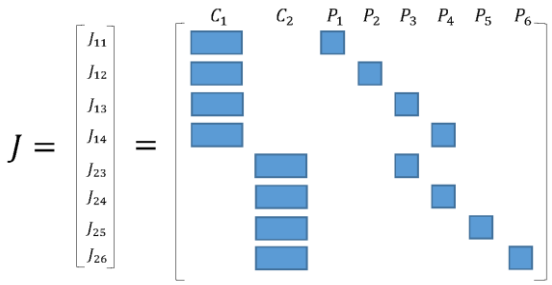


图 4-4 两个位姿和 6 个路标情况时 J 矩阵表达式

由 J 矩阵可以得到 H，如图 4-5，更直观地展示 H 的稀疏性。

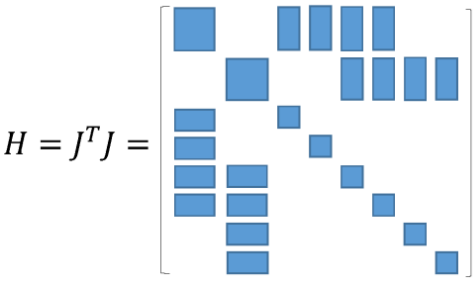


图 4-5 H 矩阵的稀疏性

从图 4-6 中可以看出 H 结构类似，非对角线上的非零元素对应着一个位姿 ξ 和一个路标点 p，共有 8×8 个矩阵块，设相机位姿 m 个，路标点 n 个，真实场景中路标点的数量一般是大于相机位姿的数量。

最后通过稀疏矩阵的求解，一般采用 Schur 消元法，Schur 消元法的过程如下：

1、将 H 矩阵方程 $H\Delta x=g$ 分解为 4 个部分，如式 4-47：

万方数据

$$\begin{bmatrix} B & E \\ E^T & \mathrm{C} \end{bmatrix} \bullet \begin{bmatrix} \Delta x_c \\ \Delta x_p \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix} \tag{4-47}$$

2、对线性方程组进行高斯消元，目标消除 E 这个小分块，得到公式 4-48：

$$\begin{bmatrix} B - EC^{-1}E^T & 0 \\ E^T & \mathrm{C} \end{bmatrix} \bullet \begin{bmatrix} \Delta x_c \\ \Delta x_p \end{bmatrix} = \begin{bmatrix} v - EC^{-1}w \\ w \end{bmatrix} \tag{4-48}$$

3、消元后，方程组变成一个只和 Δx_c 有关。于是化简可以得到位姿增量方程，如式 4-49，计算得到 Δx_c 后，再带入原方程则可以求得 Δx_p 。

$$\left[B - EC^{-1}E^T\right]\Delta x_c = v - EC^{-1}w \tag{4-49}$$

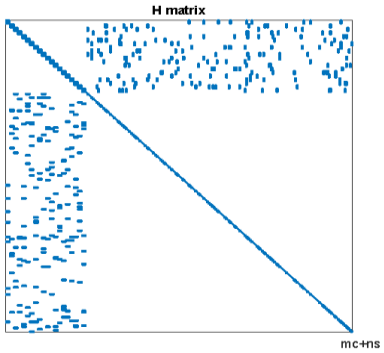


图 4-6 实际情况下的 H 矩阵

4.3 实验结果及分析

为了验证光速平差法的非线性优化性，本文选择了部分实验观测数据如表 4-7，表中展示了三维相机的位姿信息，观测点的个数，还有三维相机运动过程中对应观测点的像素坐标。

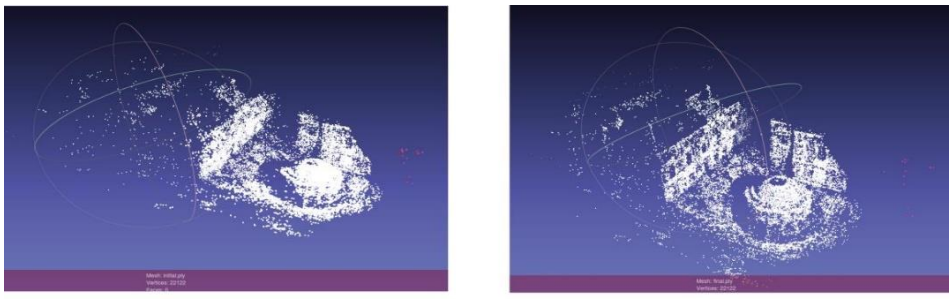


图 4-7 状态估计优化实验结果

通过图 4-7 发现增添了光速平差算法的数据，在计算更多时刻的视觉里程计的位姿和路标点后能够在全局模式下得到更为精确的表达。

万方数据

表 4-1 部分路标点和像素坐标

三维相机位置	路标点	像素坐标
0	0	-3.859900e+02 3.871200e+02
1	0	-3.859900e+02 3.871200e+02
2	0	-6.679200e+02 1.231100e+02
3	0	-5.991800e+02 1.231100e+02
9	0	-7.204300e+02 3.143400e+02
15	0	-1.151300e+02 5.548999e+02
…	…	……

4.4 本章小结

本文首先研究了视觉 SLAM 状态估计环节，传统算法使用的是扩展卡尔曼滤波进行最大后验概率估计，发现问题在于扩展卡尔曼滤波器解决非线性化时的泰勒展开式的第一项，只能用来描绘较近工作点的非线性情况。对于较远工作点，虽然此时后验概率依旧有效，但是扩展滤波器这种小范围线性近似方法，无法适用于很远的地方，极容易产生非线性误差。

研究了视觉 SLAM 状态估计概率模型，梳理清楚了卡尔曼滤波器内在原理，在马尔科夫假设基础上利用光速平差法，在新的估计点重新进行非线性优化，将一阶或二阶梯度下降估计灵活运用到新的状态估计点处，让整个优化适用范围更广。同时结合稀疏性，可以优化整个状态估计的速度。

万方数据

第五章 基于关键帧的词袋回环检测算法

前面几章主要介绍了前端视觉里程计和状态估计优化算法。视觉里程计提供特征点的提取和轨迹、地图的初值，同时还负责对所有的数据进行优化。由于视觉里程计在每一个时间点产生的累积误差问题，在长时间的累积下会使得整个SLAM会出现严重的判别失误。长时间的状态估计的错误会导致最终无法实现一个全局最优的轨迹地图。所以 SLAM 中往往还有一个单独的线程进行后端数据处理，用来解决这个累积误差，这就是回环检测算法。

回环检测是利用视觉里程计对两个非连续时间点机器人所到过的地方，根据运动状态进行数据关联的判断，从而去除累积偏移和误差量一种算法结构。如图 5-1 所示，需要将左图的累积误差消除，从而获得右图所示的正确的回环轨迹图。

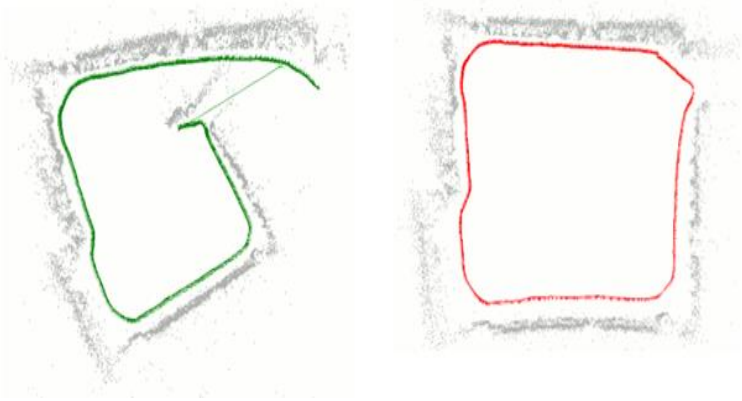


图 5-1 回环检测对累积误差的消除

回环检测是一种运动状态的量化联系，是不同时间点对同一场景的记忆重现，就像人对曾经去过的地方有种相识的感觉。对于回环检测，文献[41]中讲述了回环检测在当前 SLAM 发展的一个重要性，文献[42][43]中介绍了回环的存在是一种降低创建地图过程累积误差的绝佳手段，同时对鲁棒性的提升也是巨大的。文献[44]在全局最优的问题也解答了回环检测的限制，将后端系统的优化数据源源不断送给前端，让这个架构实现最优。

回环检测主要需要攻克 3 个问题，第一个是感知上的分歧，就是两个相似的观测信息由于环境光线或者其他因素导致可辨性受到制约，及其容易判断的错误，第二个就是前几章都分析过的数据规模，随着地图的创建，数据规模也会越来越大，运行时间会随着地点数量的增多而增长，所以逐帧检测显然是不行的。第三个对错误的回环检测的修正，不让漂移误差影响后续优化的收敛性，由此回环的匹配准确度也是一个重要的问题。

万方数据

5.1 词袋算法

5.1.1 特征点字典创建

通过前面几章的介绍能够发现三维视觉 SLAM 中的视觉里程计会不断去寻找图像的特征点，而保存每一幅图像的特征点显然是不现实的。所以本文需要采用利用其它手段来把这幅图像的特征点保存下来。经典做法是将特征点转化为类集单词，然后存储为字典的形式。字典由很多单词组成，而每一个单词代表从图像上取出来的特征点，通过机器学习的方法将各个图像的特征点分别形成点集类。

对于分类问题，如今可以使用深度学习，但是对于建筑抹灰机器人这种工业项目，需要的是简洁和稳定。于是本文采用快捷稳定的无监督聚类学习的方法。通过无监督的聚类将各个图像的特征点分为各个点集，最后形成字典。

首先，本文假设对大量的图像提取了特征点，比如说有 N 个。现在需要找一个有 k 个单词的字典，每个单词看作为局部相邻特征点的集合。这里本文采用了经典的 K-means[45]算法来解决。

本文将 K-mean 结合到图像算法里来，它的算法原理就是当有 N 个图像帧的特征点，要将图像特征点集归成 k 个类，主要有以下几个步骤：

1、随机选取图像特征点集中 k 个中心点：$c_1 ... c_k$。

2、对每一个图像特征点集，计算它到每个图像特征点集中心点之间的距离，然后统计出最小邻近点集，最后再把这个图像特征点集归为该中心一类。

3、再次计算每个图像点集类的中心点。

4、进行迭代运算，实现算法收敛，让每个图像点集之间中心点都变化很小，最后训练结束。

当然如果只是采用这种方式，仍然是一种暴力的解决方案，考虑到通用性，于是本文在 K-means 算法的基础上进行优化，对字典建立 k 叉树的层次结构，定义这个 k 叉树的深度为 d，分叉支都为 k，则 k 叉树的层次结构建立如下：

1、首先在根节点处。使用 k-means 算法将图像点集分成 k 类，构建出 k 叉树的第一层。

2、构建完第一层节点后，再次利用 k-means 算法将节点的图像点集再次聚成 k 类，构建下一层层次结构。

3、循环上述步骤，最终获得最后一层叶子结点。

5.1.2 改进的聚类特征字典相似度计算

从部分实验图集中，如图 5-2 出现了两个类似的观测图像的感知混淆问题，不

万方数据

同光照下的物体呈现出了不同灰度分布，计算机会认为不一样，但实际上两个成像场景其实是一样的，获取到的图像会存在感知上的误差，从而导致错误的回环路径检测数据对比。

图 5-2 图像的感知歧义

这里首先重新梳理一下寻找特征点单词的算法流程，如图 5-3，对一幅图像提取到 N 个特征点，对这 N 个特征点聚类后形成特征点单词，再对这幅图像以单词为基底，在字典空间中形成一个独特的列表分布，根据单词列表分布建立 K 叉树，得到最后的字典。

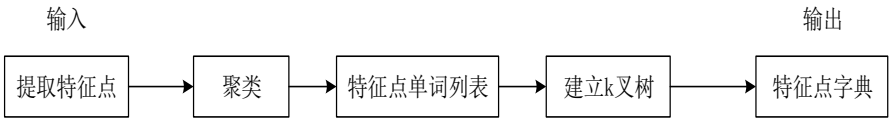


图 5-3 特征点构建字典算法流程图

传统算法问题在于对所有特征点值都不加区分的进行了单词创建，比如提取一个狗的特征构成一个字典，对于狗的毛发部分的特征是出现频率较高的单词，而鼻子部分或者其让小局部区域所构成的特征点单词出现频率低，可是往往这些位置对于分类的贡献程度更高。显然我们需要对特征点的重要性和对分类的贡献程度做出不同程度值的判断，并根据这些贡献给出不同的权重。

在文本的查重检索领域中，通常采用 TF-IDF (Term Frequency -Inverse Document Frequency)[46][47]算法。TF 的核心思想是统计一篇文章中出现较高的单词，频率高的单词往往去区分性也较高。IDF 的核心思想是统计字典中权重词汇出现的频率，出现频率最低的词被认为是区分贡献度最高的特征单词。

将文本检测算法结合到图像特征点的搜索中，原方法变为某一图像中特征点单词 w_i 出现了 n_i 次，而图像中一共出现的特征点数为 n，那么此时 TF 表示某个特征在单幅图像中出现的频率：

$$TF_i = \frac{n_i}{n} \tag{5-1}$$

同时在建立字典的过程中，继续将图像特征点单词 w_i 中的特征数量相对于字典中的特征数量的比值当作 IDF。这里定义字典中全部特征数量为 n，那么当 w_i 的特征数量为 n_i，此时这个单词的 IDF 为：

万方数据

$$IDF_i = \log \frac{n}{n_i} \tag{5-2}$$

于是， w_i 的权重为 TF 与 IDF 的乘积：

$$\eta_i = TF_i \times IDF_i \tag{5-3}$$

结合了 TF-IDF 思想后，重新定义字典的构成算法流程，如图 5-4 所示。

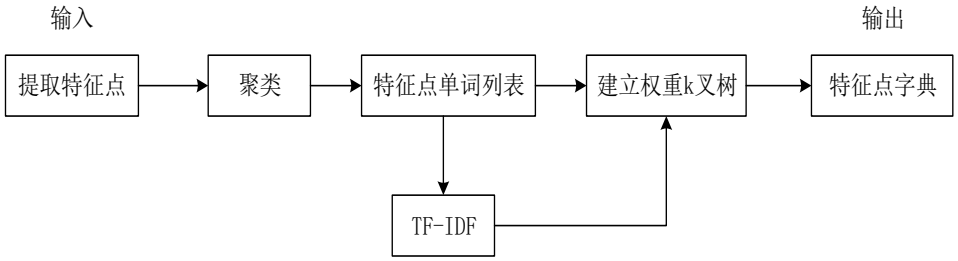


图 5-4 TF-IDF 特征点字典创建流程图

结合权重考虑后，此时给定的一幅图像 A，图像特征点可以对应到多个单词，众多单词构成了新的词袋规则。如公式 5-4 式。

$$A = \{(w_1, \eta_1), (w_2, \eta_2), ..., (w_n, \eta_n)\} \approx v_A \tag{5-4}$$

通过特征聚类后，相似的特征点会归为一个类集，所以实际上 v_A 会有很多零存在，而非零部份是该幅图像中含有的单词，这部分单词的值含有该图像中所对应的 TF-IDF 值。

最后本文利用 $L1$ 范数如公式(5-5)所示，对给定两幅图像 A 和 B 以及所对应的向量 v_A 和 v_B 计算它们的差异，差值越小则表示图像 A 和 B 越相似。

$$s(v_A - v_B) = 2\sum_{i=1}^{N} |v_{Ai}| + |v_{Bi}| - |v_{Ai} - v_{Bi}| \tag{5-5}$$

5.2 改进的关键帧词袋回环检测算法

贝叶斯视觉特征比传统算法在相同时间内提取到了更多描述子，所以需要将传统算法框架进一步优化。本文使用一帧图像来代表某一区域，也就是使用关键帧，来提高回环路径检测的速度。

文献[49]提出了关键帧的提取的两种主流方法，分别是固定间距提取法，还有基于重叠视觉提取算法。

固定间距抽取首先依据视觉里程计运动速度，按照一定距离阈值进行图像帧的提取，这种方法并不适用于场景有较多的特征点的情况，例如机器人进行旋转运动时，固定间距提取就会丢失很多关键信息，导致关键帧之间的精确估计丢失。

万方数据

基于重叠视觉算法提取是将第一帧图像暂时定义为关键帧，然后进行一个逐帧的一个特征点数的对比测量，如果测量结果小于阈值，那么就把此时的图像帧作为这一区域的关键帧。通过上述步骤就能实现关键帧的更新与检测。重叠视觉提取的主要问题在于相同场景较多的环境，会出现选取的关键帧数量少的情况，从而丢失较多关键信息。

本节首先结合传统的两种关键帧提取思想，如图 5-5，利用两种组合关键帧提取算法改进了词袋提取流程。关键帧提取完毕后建立字典，通过词袋搜索进行回环检测。

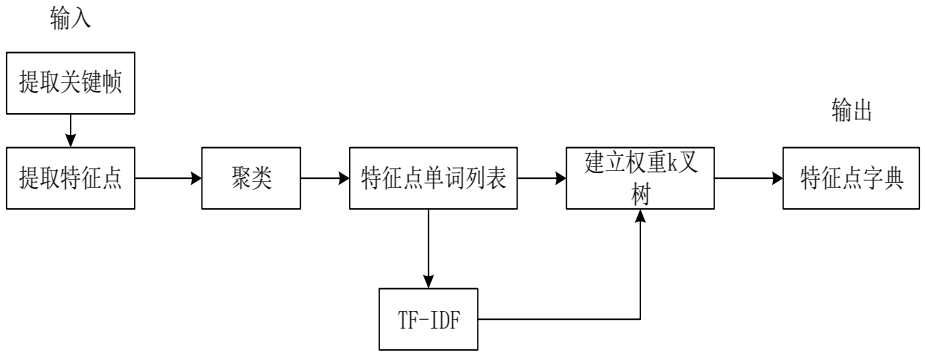


图 5-5 基于关键帧的词袋创建流程图

5.2.1 基于信息熵的关键帧提取技术

本节主要介绍基于信息熵率的关键帧算法，将信息熵率融入到传统的关键帧提取算法里面。信息熵率算法[52]可以快速对关键帧进行初判断，这对于后续的算法改进十分有利。通过信息熵的比对能够过滤部分相差较大的图像帧，然后计算贝叶斯网格统计算法匹配特征点对。最后利用帧间欧氏距离采取每隔一定间距的方式提取关键帧。提取的关键帧再进行字典的创建以及后面地图的构建。

本文先介绍信息熵率算法，信息熵指代的是信息源中所含信息元素高低的一个均值，它可以作为判断信息价值的指标。一般定义信息熵的公式如 5-6 所示：

$$S = E\left[\log\frac{1}{p(a_i)}\right] = -\sum_{i=1}^{q} p(a_i)\log(p(a_i)) \tag{5-6}$$

上式， $p(a_i)$ 代表事件 a_i 出现的概率。

依据信息熵的定义，在图像运动变换时，图像灰度运动的聚集性质其实也符合信息熵的内部含义，于是可以将局部极值运用进的灰度运动位置之中，所以本文可以将信息熵结合到图像，利用图像帧之间的信息熵的变化，给与一些预判关键帧的思路，定义如下：

万方数据

$$S=-\sum_{k=1}^{M}p(x_k)\times\log(p(x_k)) \tag{5-7}$$

$$p(x_k)=\frac{num(x_k)}{m}\times n \tag{5-8}$$

$$num(x_k)=\sum_{i}^{m}\sum_{j}^{n}s(i,j) \tag{5-9}$$

上述公式中x_k表示像素的灰度值，$p(x_k)$表示各个灰度值出现的概率，m,n分别表示图像的长度和高度，$num(x_k)$表示为灰度值为x_k时像素的个数。

因为熵的值容易同场景环境的改变而变化，通常定义一个通用的阈值会存在场景单一，不能广泛使用的情况。但是通过实验观察，本文得出帧间的熵率α是一个很好的比对参数，α的定义如公式 5-10，如果两帧之间的信息熵率的比值越大，那么图像灰度分布情况会存在一个较小的差异，对应的两幅图像息也会产生较小的差异。相反，如果两帧之间的信息熵率的比值越小，那么两幅图像的灰度分布会存在较大的差异，对应的两幅图像特征信息会是较大的差异。

$$\alpha=\begin{cases}\dfrac{s_i}{s_j} & (s_j>s_i)\\[2ex] \dfrac{s_j}{s_i} & (s_i>s_j)\end{cases} \tag{5-10}$$

在关键帧提取的过程中，图像帧间熵率的计算公式为$\alpha=\frac{s_t}{s_k}$，公式中s_k为当前关键帧的信息熵，s_t为此时待检测的帧的信息熵。如果$s_k>s_t$，反之$\alpha=\frac{s_k}{s_t}$。通过对α阈值设定就能获取到基于帧间熵率的关键帧。

5.2.2 基于信息熵的关键帧词袋构建

图像帧间熵率这种算法可以较快的进行关键帧的初步提取。虽然它可以表现出关键帧的信息，但是也会存在场景不一致，灰度分布可能相同的情况，此时图像帧间熵率的算法显然无法区分提取的，同时还会增加重复的关键帧，影响后续的建图效率。所以我们还需要将这种算法统特征点数比对，欧式距离词袋算法进行融合。算法重构步骤如下：

1、采用图像之间信息熵作区分。首先通过视觉里程计采集出一帧图像，此时我们把这一帧图像假设为这个区域的关键帧，然后计算这一帧图像和下一帧图像的一个灰度信息熵值。通过熵值的比对，我们可以初判断出是否为重复帧，如果通过阈值判断，则进入欧式距离词袋分析，反之则进行贝叶斯网格特征点数校验。

万方数据

2、采用贝叶斯视觉特征点数的比对。在第一个图像信息熵率比对环节中，会出现灰度分布相似的不同场景的图像帧，则我们需要用贝叶斯网格统计算法进行提取误匹配点的去除，然后进行贝叶斯视觉特征点对数的比对，来进一步校验是否为重复的关键帧。如果发现特征点数比对，差异较大，则说明寻找到了下一个关键帧候选，于是再利用欧式距离进行判定是否提取到这个候选的关键帧。

3、应用欧氏距离间隔采样。为了进一步达到精确性，考虑到特征点数比对可能还是会出现重复帧，于是我们在结合经典提取关键帧的算法进行运动估计的考量，确保必须在一定范围内，这个范围不光是平行移动，还包括旋转移动，通过上两个步骤校验通过的才为关键帧。，反之则反馈信号给视觉里程计继续采集下一帧图像。如果欧式校验也通过，则寻找了关键帧，然后将关键帧传递给词袋进行分析。

具体回环检测算法流程如图 5-6。

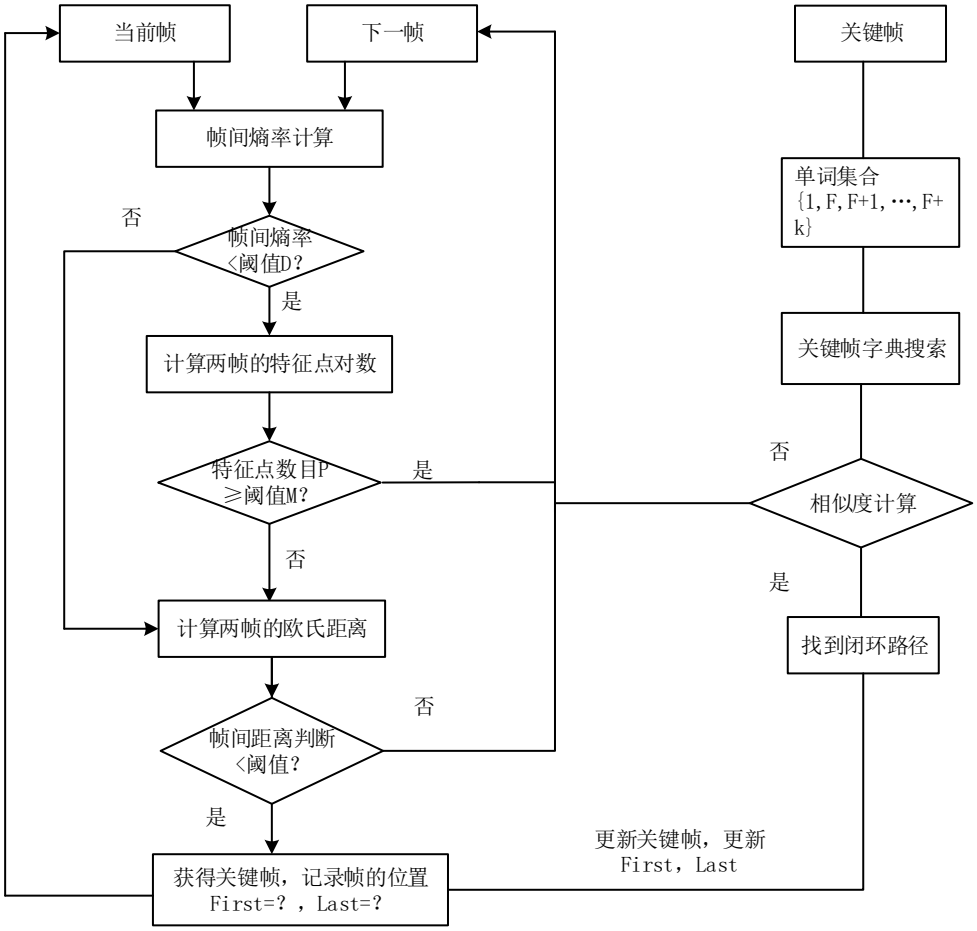


图 5-6 关键帧提取算法流程图

本节使用帧间熵率算法改进关键帧的回环路径检测算法，计算帧间熵率和帧间特征点对数，结合欧氏距离提取出关键帧，建立关键帧的词袋，最后利用字典作校验，得到回环路径的匹配。如图 5-7 所示为回环路径检测改进方案流程：

万方数据

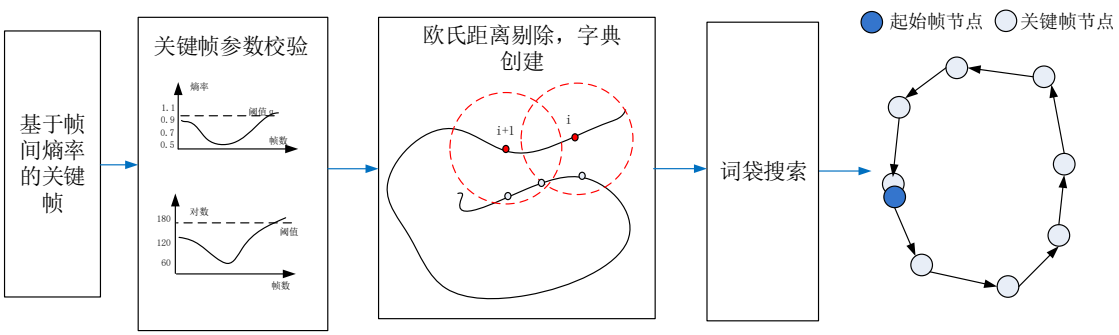


图 5-7 回环路径检测流程框图

5.3 实验结果与分析

分别应用三种方法对一段轨迹提取关键帧，如图 5-8 所示。基于信息熵率设定的阈值为 0.9；基于特征点对数的阈值为 500。需要注意的是所有算法统一距离间隔阈值设定为 0.3m，最大运动旋转度数为 35°。

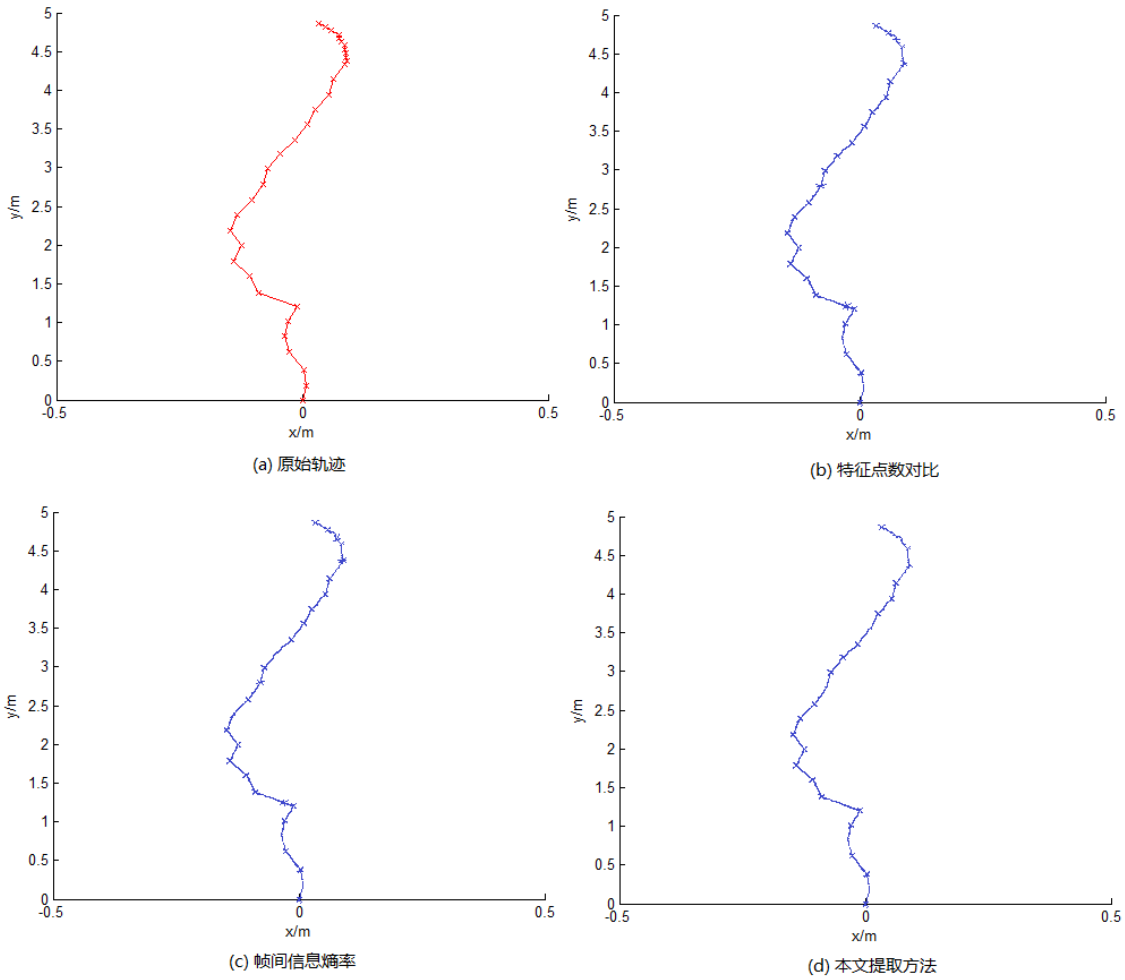


图 5-8 教研室某段关键帧提取实验结果

万方数据

从图 5-4 中的轨迹看，各个方法的图示效果差别不大，接下来我们从具体提取的关帧数分析，如表 5-1 所示。

表 5-1 教研室某段关键帧提取实验结果对比

提取方法	特征点对数	帧间熵率	两者结合使用
关键帧数	25	35	27
提取率	68.5%	88.3%	77.9%

从表 5-1 可以看出，单独使用基于帧间熵率的关键帧提取能够将环境帧数进行初步的降重，获得的关键帧数量大于单独使用特征点数对比算法。而特征点对数对比算法过滤掉了过多的关键帧，容易将不同场景特征点对数或者旋转场景的关键帧看成一致而丢失部分关键帧。

本文将两者比较方法进行了重构，在提取率上有了一定的改进。通过帧间熵率的比较来判断是否跳出特征点对数的对比，这样就可以减少关键帧的丢失，同时在利用特征点对数又能够减少帧间熵法的关键帧冗余问题，避免局部相似信息的重复提取，提高了关键帧提取效率。

最终得到如图 5-9，图 5-9 是实验图集中的一个关键帧构成的回环轨迹图。从图中可以发现基于视觉特征和信息熵能够完成回环检测轨迹的校验。

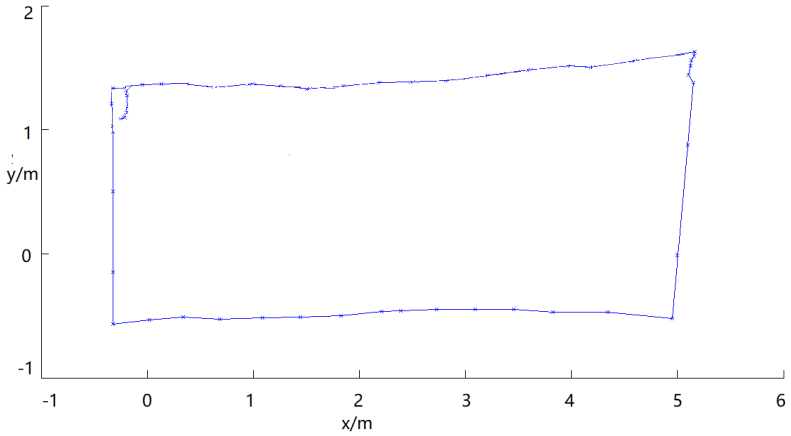


图 5-9 基于帧间熵率的回环轨迹实验结果

5.4 本章小结

本章首先研究了基于词袋的回环检测算法，明确了回环检测对于视觉 SLAM 的重要性，解决了词袋回环检测遇到的感知上的图像分歧问题。同时针对关键帧对于回环检测效率的影响。本文决定重构视觉 SLAM 的回环检测算法，本文分析了原有的关键帧提取算法，将算法结构顺序进行了一定调整，增加了信息熵的比较环节，利用信息熵对两帧之间的熵率进行对比以及关键帧的初提取。通过增加信息熵

万方数据

的比较，优化了关键帧提取的效率和准确度，加快了关键帧在词袋模型中的比对速度。

万方数据

第六章 基于三维视觉的 SLAM 技术的实验对比和分析

6.1 实验评估问题和测试数据集

本章主要对第三、四、五章进行一个实验结果和分析的汇总，验证整个算法的合理性和可行性。主要完成三个问题的验证和分析：

1、贝叶斯网格统计模块对于传统 SLAM 算法上时间和鲁邦性的提升，主要对比 SIFT 和 ORB 特征点算法，以及特征匹配筛选方法 BF，FLANN，RANSAC。

2、在三维视觉 SLAM 状态估计环节，传统算法使用的是扩展卡尔曼滤波进行最大后验概率估计，本文使用了光速平差法，验证光速平差法对于非线性运动估计的优化。

3、在三维视觉 SLAM 回环检测算法中，验证算法重构后的正确性，能够解决感知上的分歧问题，还有数据规模问题。随着地图的创建，数据规模也会越来越大，最后能够对错误的回环检测的修正，不让漂移误差影响后续优化的收敛性。

在对三维视觉 SLAM 算法评估中，为了实验数据对比的公平性，本文使用了德国慕尼黑工业大学提供的 TUM 数据集和 NYU V2 数据集进行测试结果的对比和评估。TUM 数据集中包含了各种实验场景图，一共有 3141 组实验图像，其中轨迹数据是在高速运动中使用了 8 个 Kinect 相机采集得到的位姿信息结果。同时旋转，平移的图像对组是作者通过手持 Kinect，在缓慢运动拍摄得到，保证了数据的准确可靠性。而 NYU V2 数据集中包含了 408000 幅 RGBD 图像，一共有 1449 类带标签的目标场景，图像主要来自室内空间，同样也是由 Kinect 相机拍摄得来。

本文会使用到 TUM 和 NYU V2 官方测试工具，其中官方工具用来测试模型误差，也就是通过采用 SLAM 技术建好的三维地图和官方的标准数据集的三维地图进行运动关系序列的评估，评估工作主要是旋转误差 Re 和平移误差 Te，通过两个误差计算出轨迹损失。

其中旋转误差 Re 和平移误差 Te 定义为：

$$\begin{cases}\theta=2\cos^{-1}(q,\hat{q}) \\ \mathrm{Re}=\min(\theta,2\pi-\theta)\end{cases} \tag{6-1}$$

公式 6-1 中，$\hat{q}$ 为运动旋转关系矩阵，q 为官方轨迹对应的运动旋转关系矩阵。

$$Te=\sqrt{(t'_x-t_x)^2+(t'_y-t_y)^2+(t'_z-t_z)^2} \tag{6-2}$$

公式 6-2 中，(t'_x,t'_y,t'_z) 为运动平移关系矩阵，(t_x,t_y,t_z) 为官方轨迹对应的运动

万方数据

平移关系矩阵。

6.2 特征匹配算法对比和分析

本节选取了测试数据集 TUM 中一共 500 对图片以及 NYU V2 数据集中 789 连续时刻图片进行了特征提取和匹配实验，对特征描绘子提取的数目，以及时间，匹配情况做了平均统计对比。图 6-1 为测试集中两个场景的一个实验结果展示。左右图分别为 SIFT 算子和 BGS 算子提取与匹配的一个对比，上下图分别为 SIFT 算子和 BGS 算子经过 RANSAC 算法筛选后的匹配对比。

图 6-1 特征提取和匹配筛选算法后的对比

通过图 6-1 的一个对比，可以发现 BGS 算法从视觉上的匹配效果更好，这里要注意一个细节，实验中 BGS 算法采用的是 ORB 提取的特征点，然后对这些点进行了贝叶斯网格统计。具体 SIFT、ORB、BGS 算法给特征提取与匹配带来的对比，见表 6-1。这里注意的是为了比较时间效率的提升，本文控制了其他变量，也就是 ORB 算法和 BGS 算法的提取数目控制在了 500 以内。这样能展现出相同的特征提取上，BGS 算法在时间效率上的改进。

表 6-1 特征点匹配算法实验结果对比

特征点提取算法	特征点提取时间(ms)	特征点匹配时间(ms)	特征点数目	匹配点对数目	总时间(ms)
SIFT	65	10	350	141	140
ORB	12	13	398	123	37
BGS	12	1.5	398	148	25

从表 6-1 我们可以得出，在单幅图像的平均特征提取时间上，ORB 和 BGS 整体上的提取速度是 SIFT 的 5 倍。而对于单个特征点平均提取时间，SIFT 算子平均每一个特征点的提取时间为 0.3651ms，ORB 算子平均每一个特征点的提取时间为

万方数据

0.03，而 BGS 采用了 ORB 算法提取环节，所以在特征提取的时间表现上是一致的。我们可以看出平均每一个特征提取时间上，以 ORB 提取为核心的算法比 SIFT 提取算法快了 100 倍，这也是 SIFT 算法无法应用状况的原因，就是效率太慢。特别的，在 ORB 和 BGS 的特征匹配时间上，由于网格统计的优化，BGS 的匹配时间只需要 1.5ms，快过 ORB 的 13ms 时间。

但是在匹配率上，SIFT 准确率为 93.26%，ORB 准确率为 61.81%，BGS 的准确率为 74.37%。可见在匹配率上 SIFT 仍然占据着最高的位置。

总的来说，由于贝叶斯网格的添加，让 ORB 算子在匹配中展现了惊人的效率，只需要 1.5ms 的时间，大大提高了整个算法匹配的速度。由此，通过实验的验证我们将 BGS 应用到三维视觉 SLAM 中对于实时性和鲁棒性都是有提高的。

上面分析了本文涉及到的特征点提取和匹配算法，下面本文对匹配点筛选算法进行一个分析总结，注意这里为了比对界限更明显，表中前三项统一使用 BGS 的提取匹配方法，同时去除了特征点数目的限制，方便在大匹配数目时对筛选算法的一个比对。为了显示一个 BGS 模块的优越性，又加入了传统的 ORB 和 RANSAC 算法的时间对比。

表 6-2 视觉匹配筛选算法实验结果对比

匹配筛选算法	匹配点数目	匹筛选后的对数	总时间(ms)
BGS+BF	1124	654	54.725
BGS+FLANN	1124	479	30.474
BGS+RANSAC	1124	525	28.131
ORB+RANSAC	1124	435	35.325

从表 6-2，可以看出 BF 的匹配筛选的对数更多，但是耗时确是最长的，这符合暴力搜索匹配的特性。结合 BGS 模块下 RANSAC 获得的匹配筛选对数比传统的 ORB 结合 RANSAC 的匹配对数更多，同时消耗的时间也少于 ORB 结合 RANSAC 的情况，验证了网格统计对于 RANSAC 运算过程收敛性的帮助，以及贝叶斯网格对于特征的非监督聚类特性。

6.3 SLAM 非线性优化算法对比与分析

本节选取了测试数据集 TUM 中一共 500 对位姿变换的图片和 NYU V2 数据集中 789 连续时刻图片进行了状态估计和非线性优化的实验对比。得出表 6-3 的结果。注意这里我们没有限制 ORB 的算子提取数目。

从表 6-3 可以得到，基于光速平差法状态估计花费的时间比原始方法高，但是

万方数据

却比传统 EFK 状态估计误差的小，同时在实验中优化成功率也更高。

表 6-3 SLAM 状态估计算法实验结果对比

	特征点对数	传统 EFK 的状态估计		光速平差法状态估计	
		状态估计误差	状态估计时间(ms)	状态估计误差	状态估计时间(ms)
SIFT	140	0.00444218	19	0.000964601	23
ORB	435	0.0113304	36	0.000661158	42
BGS	525	0.0112301	28	0.000561132	31

本文将整个前端视觉里程计的数据统计到了表 6-4 内，其中将 ORB 算子的提取数目控制在了 500 以内，表中误差值为运动变换下的局内点同原始特征匹配点的对比。

表 6-4 视觉里程计总体实验结果对比

特征点提取方法	特征点提取数目	特征点提取时间(ms)	匹配对数目	匹配时间(ms)	优化时间(ms)	总时间(ms)	误差
SIFT	350	65	140	10	19	220	0.001426
ORB	398	12	123	13	9	41	0.000672
BGS	398	12	148	1.5	8	35	0.000561

通过表 6-4，可以看出总的前端时间消耗上来看，以 SIFT 为基础的整个状态估计耗时是最长的，达到了 220ms，也就是 1 秒能处理 4 帧图像，显然无法达到实时的要求。而对于 ORB 为特征点的视觉里程计上，总时间为 41ms，也就是一秒能处理 25 帧图像，所以这也是 ORB 现今能够运用进 SLAM 技术的原因。而对于本文的算法重构后，在状态估计消耗 8ms，比 ORB 减少了 10%的时间，总时间上消耗 35ms，对比 ORB 视觉里程计减少了 12%，每秒能处理 28 帧图像。同时误差上面，虽然仍然达不到 SIFT 的高精确度，但是也并没有丧失传统 ORB 视觉里程计的一个精确度。可见本文的方法是可行，同时在时间效率上获得了提高，在地图创建过成中，仍然能达到原有精度，可见其鲁棒性也有一定的提升。

6.4 回环检测算法对比与分析

本文选取了 TUM 数据集中某实验场景平面图，如图 6-2，移动机器人在实验场景的行走路径如下，分析关键帧词袋回环算法的提取效果，红色圆点是作者制定的几个图像采集标志位同时也是重点观测位置，本文把这几个位置的共 300 张图

万方数据

像作为字典的无监督训练图集，字典的规模取 $k=10$，$d=5$。图像中数字序号代表观测序列。实线箭头指向代表相机的朝向，虚线箭头指向代表移动方向，机器人在 A 和 B 点进行了 180° 的旋转。

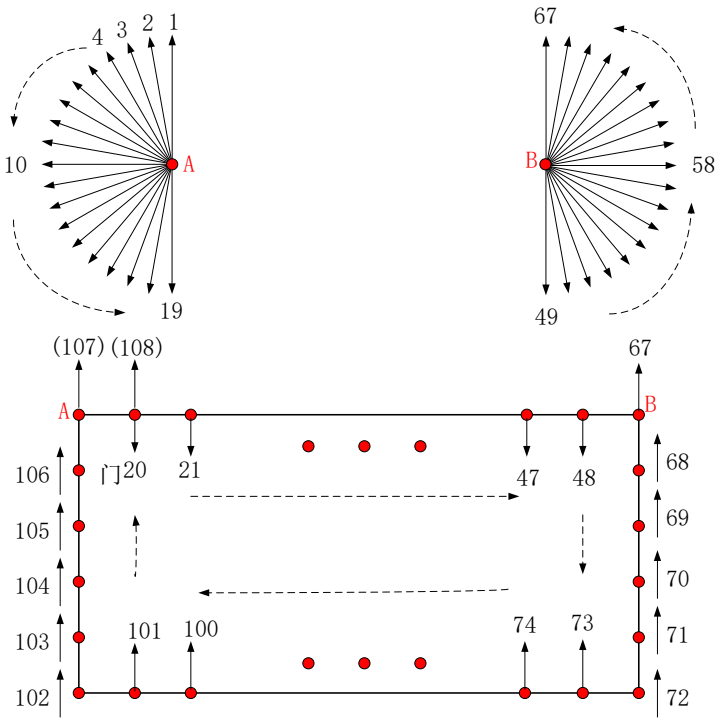


图 6-2 实验图集场景移动机器人运行轨迹

根据这个轨迹，统一设定欧式距离间隔阈值为 0.4m，旋转运动角度值为 35°，图像帧间信息熵率的阈值为 0.85，基于贝叶斯视觉特征比对数的阈值为 500。三种关键帧提取的结果分别如下所示，图 6-3，左图为基于特征点数词袋回环检测结果，右图为基于帧间熵率的回环检测结果。图 6-4 为本文重构后的算法结果。

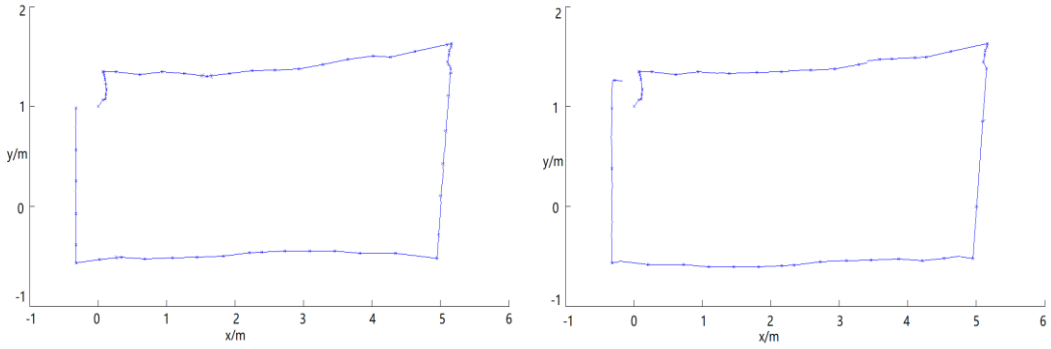


图 6-3 基于词袋的回环检测算法实验结果

万方数据

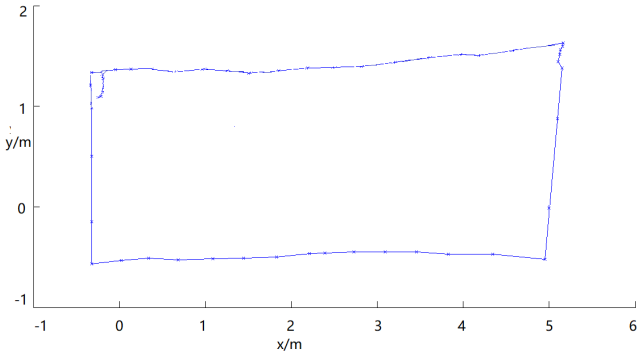


图 6-4 本文的回环检测算法实验结果

表 6-5 实验图集场景关键帧提取个数

提取方法	特征点数比对	图像信息熵率	两者结合
关键帧数	304	324	313
提取率	73.6%	79.8%	76.9%

根据图 6-3，6-4，表 6-5，能够看出本文的关键帧的词袋算法在回环处获得了更为精确的表达。只是采用特征点数对比或者欧式距离分割，在回环处很容易导致关键帧的丢失。通过本文回环检测算法的重构后可以获得更加稳定的关键帧提取效果。完成上述实验现象，本文还从 TUM 和 NYU V2 中选取了 30 个场景进行了实验测试，并将测试相关数据汇总到了表 6-6。

表 6-6 回环轨迹误差统计对比

	X 方向偏差/m	Y 方向偏差/m	位置误差/m	转向角误差/rad	均方根最大误差/m	时间/s
EFK-SLAM	0.8264	2.2437	2.4431	0.6754	2.7831	90.6784
光速平差 SLAM	0.6571	1.4013	1.6467	0.4826	1.9859	88.7631

从上述各图及表 6-5 和 6-6 中可以看出，三种方法提取情况虽然类似，但是在回环处的检测上，情况却不同，本文的方法在回环处检测到了更多的关键帧，没有导致信息的丢失，在数据集的表现上，各方向偏差都有一定的减小，其中位置偏差比较 EFK-SLAM 算法减少了百分之三十，充分体现了利用全局时间路标信息的优越性。虽然光速平差法在耗时上有所增加，但是结合重构的关键帧算法，可以将时间复杂度进行一定的简化，同时误差率相比下降很多。在均方根误差丢失对比上，光速平差法 SLAM 的回环检测上也显得更小，说明采用光速平差法配合关键帧的词袋算法能够获得更加稳定的效果，在回环处的地图创建效果更好。

万方数据

6.5 三维地图创建结果对比与分析

根据本文的三维视觉 SLAM 技术框架，利用 TUM 数据集中的 860 帧彩色和深度数据，完成了一组三维地图创建，图 6-5，左图为视觉里程计下的结果，右图为非线性优化后和回环检测后的的结果。从图中我们能够发现，前端视觉里程计获得了大量的特征信息，后端回环检测环节里面消除了大部分的错误的回环约束，在三维地图创建的过程中没有发生的大规模的变形和扭曲。

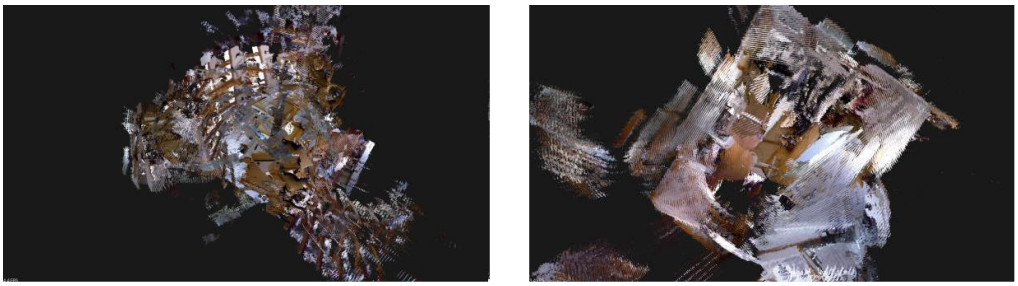


图 6-5 三维地图创建实验结果

按照本文的方法，本文将 TUM 数据集中的 30 个场景和 NYU V2 中的 20 个场景进行了重建，并将全部点云地图送入了 TUM 官方的测试工具和 NYU V2 官方的测试工具，同官方标准的三维地图轨迹进行一个误差对比分析，同时将其他算法也进行了一个对比，如表 6-6 表中斜杠的部分是没有反馈数据的，原因为 MonoSLAM 为单目相机下的解决方案，基于单目相机的 SLAM 无法计算绝对深度，也就是无法获得机器人运动轨迹以及地图的真实大小。而 RatSLAM 核心是按照环境模板神经网络的训练，定点匹配，所以无跟踪损失数据。

表 6-6 官方数据集实验结果对比

	室外轨迹 1 长度 413m			室外轨迹 2 长度 438m			室内轨迹 3 长度 413m		
SLAM 技术方案	轨迹误差 (cm)	跟踪损失	内存使用量 (MB)	轨迹误差 (cm)	跟踪损失	使用内存量 (MB)	轨迹误差 (cm)	跟踪损失	内存使用量 (MB)
MonoSLAM		95.7%	73		90.6%	646		97.3%	102
PATM	33.4	7.6%	1543	24	15.9%	718	23.4	3.5%	437
ORB-SLAM	12	33.9%	5537	11.2	6.5%	2089	10.1	0.0%	4222
LSD-SLAM	38.8	0.1%	2728	27.6	12.0%	1376	15.1	78.6%	1067
RatSLAM	37.4		402	24.4		444	17.9		333
本文方案	8	25.5%	5673	9	3.2%	2231	7	0.2%	4332

万方数据

通过表 6-6 可以发现，本文方案虽然在实时内存占用上是最多的，原因在于本文使用了更多的匹配点，但是轨迹误差表现上同 ORB-SLAM 有一定提升的。当然 LSD-SLAM 作为稠密三维地图的代表，其跟踪损失是最低的。通过表格数据能够看出，本文方案，在实时提高性的情况下，鲁棒性并没有损失，验证本文方案的正确性和可行性。

万方数据

第七章 总结与展望

7.1 本文研究总结

三维视觉 SLAM 技术作为一种高新科技正逐步应用到各个领域之中。本论文依托项目基于三维视觉的 SLAM 技术的高压电力线检测和建筑抹灰机器人，对三维视觉 SLAM 技术做了深入研究。高压电力线检测，涉及无人机的一个飞行巡检，不光需要 GPS 的一个定位，同时还需要一个三维场景的即时创建和定位来进行避障和精确位置修正。同时对于建筑抹灰机器人，一个好的三维地图实时创建，让建筑抹灰机器人能够完成自主定位移动，主动避让，同时还能自适应根据不同墙体墙角进行自动抹灰。本文针对在实际项目中所遇到的问题，对重要环节的算法进行了改进和优化，主要发现以下问题：

在三维视觉 SLAM 算法中，经典的 SIFT 特征提取算法在工程应用上是无法使用的，原因在于 SITF 的特征点向量维数导致高计算复杂度，严重制约了系统的实时性。目前视觉里程计中的特征提取方法大多数是 ORB 特征提取算法，通过将特征点向量维数降低了一半，虽然达到了实时的使用效果，却因此丢失了部分精度，在质量差和纹理低的场景中，匹配效果较差。如果增加 ORB 的特征提取点数，又会导致后序匹配筛选以及计算运动关系速度降低。

在三维视觉 SLAM 状态估计环节，传统算法使用的是扩展卡尔曼滤波进行最大后验概率估计，问题在于扩展卡尔曼滤波器解决非线性化时的泰勒展开式的第一项，只能用来描绘较近工作点的非线性情况。对于较远工作点，扩展滤波器这种小范围线性近似方法，无法适用于很远的地方，极容易产生非线性误差。

在三维视觉 SLAM 回环检测算法中，第一个问题是感知上的分歧，也就是两个相似的观测信息由于环境光线或者其他因素导致可辨性受到制约，极其容易判断的错误，第二个就是数据规模，随着地图的创建，数据规模也会越来越大，存储计算要求也越来越大。第三个就是对错误的回环检测的修正，不让漂移误差影响后续优化的收敛性，由此回环的匹配准确度也是一个重要的问题。

对于上述问题，本文进行了全面的研究，如下：

1、研究了 SLAM 的视觉特征和匹配过程中出现复杂度高的算法环节，在传统的视觉提取基础上，本文通过建立贝叶斯视觉模型，利用网格区域的加权统计，设计出一种贝叶斯网格统计算法模块，加速了视觉特征匹配效率，提高了传统匹配算法的鲁棒性。

2、研究了三维视觉 SLAM 状态估计概率模型，梳理清楚了卡尔曼滤波器内在

万方数据

原理。在马尔科夫假设基础上，本文利用光速平差法在新的估计点重新进行非线性优化，将一阶或二阶的梯度下降估计灵活运用到新的状态估计点处。光速平差法让整个优化适用范围更广，同时结合矩阵稀疏性，优化了整个状态估计的速度。

3、研究了三维视觉 SLAM 的回环检测算法。利用非监督学习对图像特征点进行单词构建，通过词袋中字典的对比获得回环反馈。本文分析了原有的关键帧提取算法，将算法结构顺序进行了一定调整，增加了信息熵的比较环节，利用信息熵对两帧之间的熵率进行对比，通过对比结果能够发现如果两帧之间的信息熵率的比值越大，那么图像灰度分布情况会存在一个较小的差异，对应的两幅图像息也会产生较小的差异。相反，如果两帧之间的信息熵率的比值越小，那么两幅图像的灰度分布会存在较大的差异，对应的两幅图像特征信息会是较大的差异。本文通过增加信息熵的比较加快了词袋单词的比对速度，优化了关键帧提取的效率和准确度。

最终本文完成的工作有设计了基于贝叶斯网格统计的特征提取匹配算法，在传统 SLAM 算法框架中，融入贝叶斯网格统计算法，可以加速运动模型的计算，提高系统的鲁棒性。本文使用了光速平差法替代传统的 EFK 算法，获得了更好的状态估计和优化效果。同时本文对 SLAM 技术的回环检测算法进行了调整，在不影响效率的情况下，增加了特征点数量，通过基于帧间熵率的关键帧检测，快速过滤干扰的图像帧，让关键帧的提取速度和准确度得到了提高。

7.2 未来改进与展望

本文基于三维视觉的 SLAM 技术研究仍然有不足和改进的地方。比如第三章中贝叶斯网格统计模块虽然能够提高视觉匹配的速度和鲁棒性，但是在纹理低的场景表现仍然存在提取特征点少，容错率低的问题。在非线性优化算法中，关键帧的提取虽然进行了重构，但是对于部分场景仍然存在提取率不足，回环检测效果差的问题。

展望未来，三维视觉的 SLAM 技术中词袋算法结合深度学习，可以让视觉里程计不仅可以识别回环路径，还能识别事物场景类别，实现智能识别场景的功能。

万方数据

致谢

三年的科研岁月，求知路上遇到了很多帮助我的人。我的研究生导师于鸿洋教授在科研方面的热情对我触动很深，让我整个研究生的科研岁月保持着高度的专注。于老师对于新知识的接纳超出了我的想象，让我明白学无止境不分年龄。在科研路上，时常遇到磕磕绊绊，于老师的耐心指导，让我不断的克服每一个艰难险阻。这里，感谢于老师三年的指导，我会一直保持教研室的科研精神去攻克以后的每一个目标。

回忆过去的三年，这里还要感谢教研室管理王昭婧师姐，我从一个科研小白到专业的科研工作者离不开师姐的帮助。师姐不仅是学习上的小能手，还是心理疗养师，感谢师姐对我的各种帮助。三年的教研室岁月，感谢一路陪伴的同学，崔凡钦、韦维、向憧，我从你们身上获益良多。感谢小师弟小师妹们，佘倩、任亚冰、周乐天、毛伟鹏等在科研和生活上给我的帮助。

三年的生活岁月，学习路上总是有那么一群小伙伴陪你欢笑。 感谢我的成都探险小分队，云轲和梦梦大宝贝。三年里一路欢笑，感谢成都成都分队，翔子，利姐，秋阳哥等，感谢你们的爱护和支持。

最后感谢我的老爸老妈和我的家人，感谢你们日以继夜对我的关心和爱，学习生涯即将告一段落，感谢你们给予我成长的力量，面对困难的勇气，解决问题的聪慧。

祝大家身体健康，开开心心！

万方数据

参考文献

[1] Moravec H P, Elfes A. High resolution maps from angle sonar[C]// ICRA. 1985:116-121.

[2] Ayache N. Maintaining representation of the environment of a mobile robot[J]. IEEE Transactions on Robotics & Automation, 1989, 5(6):804-819.

[3] Alan W. Introduction to AI Robotics[M]. A Bradford Book : The MIT press, 2000.

[4] Castellanos J A, Tardós J D. Simultaneous Localization and Map Building[M]// Mobile Robot Localization and Map Building. Springer US, 1999.

[5] Davison A J, Reid I D, Molton N D, et al. MonoSLAM: Real-Time Single Camera SLAM[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2007, 29(6):1052.

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

[7] Scaramuzza D, Fraundorfer F. Visual Odometry [Tutorial][J]. Robotics & Automation Magazine IEEE, 2011, 18(4):80-92.)。

[8] Moravec H P. Obstacle avoidance and navigation in the real world by a seeing robot rover[M]. Stanford University, 1980.

[9] Lacroix S, Mallet A, Chatila R, et al. Rover Self Localization in Planetary-Like Environments[J]. Artificial Intelligence, 2013, 440:433.

[10] Nistér D, Naroditsky O, Bergen J. Visual Odometry[C]// Proc. IEEE Conf. Computer Vision and Pattern Recognition. 2004:I-652-I-659 Vol.1.

[11] Pupilli M, Calway A. Real-Time Camera Tracking Using Known 3D Models and a Particle Filter[C]// International Conference on Pattern Recognition. IEEE, 2006:199-203.

[12] Montemerlo M S. Fastslam: a factored solution to the simultaneous localization and mapping problem with unknown data association[M]. Carnegie Mellon University, 2003.

[13] Civera J, Davison A J, Montiel J M M. Inverse Depth Parametrization for Monocular SLAM[J]. IEEE Transactions on Robotics, 2008, 24(5):932-945.

[14] Mourikis A I, Roumeliotis S I. A Multi-State Constraint Kalman Filter for Vision-Aided Inertial Navigation[J]. 2007, 22:3565-3572.

[15] Li M, Mourikis A I. Improving the accuracy of EKF-based visual-inertial odometry[C]// IEEE International Conference on Robotics and Automation. IEEE, 2012:828-835.

[16] Klein G, Murray D. Parallel Tracking and Mapping for Small AR Workspaces[C]// IEEE and ACM International Symposium on Mixed and Augmented Reality. IEEE, 2008:1-10.

[17] Mur-Artal R, Montiel J M M, Tardós J D. ORB-SLAM: A Versatile and Accurate Monocular

万方数据

SLAM System[J]. IEEE Transactions on Robotics, 2015, 31(5):1147-1163.

[18] Engel J, Schöps T, Cremers D. LSD-SLAM: Large-Scale Direct Monocular SLAM[J]. 2014, 8690:834-849.

[19] Newcombe R A, Lovegrove S J, Davison A J. DTAM: Dense tracking and mapping in real-time[C]// IEEE International Conference on Computer Vision. IEEE, 2011:2320-2327.

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

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

[22] Rublee E, Rabaud V, Konolige K, et al. ORB: An efficient alternative to SIFT or SURF[C]// IEEE International Conference on Computer Vision. IEEE, 2012:2564-2571.

[23] Calonder M, Lepetit V, Strecha C, et al. BRIEF: binary robust independent elementary features[C]// European Conference on Computer Vision. Springer-Verlag, 2010:778-792.

[24] Mirowski P, Palaniappan R, Ho T K. Depth camera SLAM on a low-cost WiFi mapping robot[C]// IEEE International Conference on Technologies for Practical Robot Applications. IEEE, 2012:1-6.

[25] Endres F, Hess J, Sturm J, et al. 3-D Mapping With an RGB-D Camera[J]. IEEE Transactions on Robotics, 2017, 30(1):177-187.

[26] Bay H, Tuytelaars T, Gool L V. SURF: Speeded Up Robust Features[C]// European Conference on Computer Vision. Springer-Verlag, 2006:404-417.

[27] Ke Y, Sukthankar R. PCA-SIFT: a more distinctive representation for local image descriptors[C]// IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, 2004:506-513.

[28] Morel J M, Yu G. ASIFT: A New Framework for Fully Affine Invariant Image Comparison[J]. Siam Journal on Imaging Sciences, 2009, 2(2):438-469.

[29] P. J. Besl, N. D. McKay. Method for registration of 3-D shapes[C]//Robotics-DL tentative. International Society for Optics and Photonics, 1992: 586-606.

[30] D. Aiger, N. J. Mitra, D. Cohen-Or. 4-points congruent sets for robust pairwise surface registration[J]. ACM Transactions on Graphics (TOG), 2008, 27(3): 85.

[31] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions[J]. JOSA A, 1987, 4(4): 629-642.

[32] Zorzi M. Robust Kalman Filtering under Model Perturbations[J]. IEEE Transactions on Automatic Control, 2017, PP(99):1-1.

万方数据

[33] Crandall D J, Owens A, Snavely N, et al. SfM with MRFs: Discrete-Continuous Optimization for Large-Scale Structure from Motion[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2013, 35(12):2841-2853.

[34] Plett G L. Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs : Part 3. State and parameter estimation[J]. Journal of Power Sources, 2004, 134(2):1356-1368.

[35] Carlevaris-Bianco N, Kaess M, Eustice R M. Generic Node Removal for Factor-Graph SLAM[J]. IEEE Transactions on Robotics, 2017, 30(6):1371-1385.

[36] Newman M E J. Newman MEJ. The structure and function of complex networks. SIAM Rev 45: 167-256[J]. 2003, 45(2):167-256.

[37] Trott O, Olson A J. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization and multithreading[J]. Journal of Computational Chemistry, 2010, 31(2):455-61.

[38] Epperson J F. An introduction to numerical methods and analysis[M]// An introduction to numerical methods and analysis /. Wiley-Interscience, 2007:A142.

[39] Kümmerle 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.

[40] Polok L, Ila V, Solony M, et al. Incremental Block Cholesky Factorization for Nonlinear Least Squares in Robotics[C]// Robotics: Science and Systems. 2013.

[41] Li M, Lin R, Wang H, et al. An efficient SLAM system only using RGBD sensors[C]// IEEE International Conference on Robotics and Biomimetics. IEEE, 2013:1653-1658.

[42] Lategahn H, Geiger A, Kitt B. Visual SLAM for autonomous ground vehicles[C]// IEEE International Conference on Robotics and Automation. IEEE, 2011:1732-1737.

[43] Beare G. Past, present and future[J]. Indexer, 2007, 25(4):257-264.

[44] Magnusson M, Andreasson H, Nuchter A, et al. Appearance-based loop detection from 3D laser data using the normal distributions transform[J]. Journal of Field Robotics, 2009, 26(11-12):892–914.

[45] Cummins M, Newman P M. Appearance-only SLAM at large scale with FAB-MAP 2.0[J]. International Journal of Robotics Research, 2011, 30(9):1100-1123.

[46] Wang H M 1. Online mapping with a mobile robot in dynamic and unknown environments[J]. International Journal of Modelling Identification & Control, 2008, 4(4):415-423.

[47] Endres F, Hess J, Sturm J, et al. 3-D Mapping With an RGB-D Camera[J]. IEEE Transactions on Robotics, 2017, 30(1):177-187.

万方数据

[48] Stewart R L, Zhang H. Image similarity from feature-flow for keyframe detection in appearance-based SLAM[C]// IEEE International Conference on Robotics and Biomimetics. IEEE, 2011:305-312.

[49] Glocker B, Shotton J, Criminisi A, et al. Real-Time RGB-D Camera Relocalization via Randomized Ferns for Keyframe Encoding[J]. IEEE Transactions on Visualization & Computer Graphics, 2015, 21(5):571-583.

[50] Barbieri T T D S, Goularte R. KS-SIFT: A Keyframe Extraction Method Based on Local Features[C]// IEEE International Symposium on Multimedia. IEEE, 2015:13-17.

[51] Stalbaum J, Song J B. Keyframe and inlier selection for visual SLAM[C]// International Conference on Ubiquitous Robots and Ambient Intelligence. IEEE, 2013:391-396.

[52] Steinbrucker F, Kerl C, Cremers D, et al. Large-Scale Multi-resolution Surface Reconstruction from RGB-D Sequences[C]// IEEE International Conference on Computer Vision. IEEE Computer Society, 2013:3264-3271.

万方数据

攻读硕士期间取得的研究成果

[1] Chuang Zeng, HongYang Yu, Time image sequence self-encoding statistics to improve visual odometer. On behalf of the 2018 International Conference on Frontiers of Materials, Manufacturing, Mechanical Engineering.

万方数据