分　类　号：　　单位代码：10183
研究生学号：2019562013　　密　　级：公开

吉 林 大 学

硕士学位论文

(学术学位)

多传感器融合的 SLAM 设计与实现

Design and Implementation of Multi-Sensor Simultaneous Localization and Mapping System

作者姓名：高塞航
专　　业：计算机科学与技术
研究方向：自动驾驶
指导教师：王刚 教授
培养单位：人工智能学院

2022 年 5 月

多传感器融合的SLAM设计与实现

Design and Implementation of Multi-Sensor Simultaneous Localization and Mapping System

作者姓名：高塞航

专业名称：计算机科学与技术

指导教师：王刚 教授

学位类别：工学硕士

答辩日期： 2022 年 5 月 24 日

吉林大学硕士学位论文原创性声明

本人郑重声明：所呈交学位论文，是本人在指导教师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：高嘉航

日期：　　2022 年 4 月 15 日

摘要

多传感器融合的SLAM设计与实现

目前，自动驾驶作为计算机科学与汽车工业的交叉领域，拥有广阔的市场前景，迅速成为人工智能的热门领域之一。自动驾驶包含定位、感知、决策和导航等关键模块。而即时定位与建图（Simultaneous Localization and Mapping，SLAM），能够有效的结合多种传感器的优势，对车辆位姿进行估计，对周围环境进行构建，为构建包含各种路况信息的高精度地图提供关键参照。因此，准确可靠的状态估计和地图构建是自动驾驶系统的重要组成部分。

目前的主流方法只专注于静态场景任务中的几何特征配准方法。一方面，基于几何特征的配准十分容易受到动态物体的干扰，从而导致精度下降。另一方面，随着深度语义分割网络的发展，除了几何信息之外，我们可以方便的从点云中获取语义信息。语义特征可以用作几何特征的补充，可以提高里程计和回环检测的准确性。语义信息还可以过滤掉点云数据中动态物体，例如行人和车辆等等，这些动态物体会导致生成的地图中的出现残影或者基于地图的定位失败。

通过上述分析，本文基于LIO-SAM框架的前端里程计和回环检测模块进行改进，本文的主要贡献如下：

（1）本文优化了 SPVNAS 网络提供的语义信息，并将其与激光雷达前端里程计相结合。该算法在有大量运动物体的实验场景中，系统仍可以进行准确配准。

（2）通过结合语义信息，本文提出了一种语义辅助的 Scan Context 方法，并使用语义ICP对其结果进行校验，将其应用为系统的回环检测策略。该算法使得系统在回环检测时更关注静态场景的匹配，可以纠正位姿累计误差并提升全局建图效果。

（3）本文使用实验室的传感器，采集了多种场景下的吉林大学校园数据，并在经典的自动驾驶数据集 KITTI 和吉林大学校园数据上进行的了多组相关

对照试验。实验结果表明，我们的方法优于纯几何方法，尤其是在动态场景下，具有良好的泛化能力。

关键词：

人工智能，自动驾驶，即时定位与建图，语义分割，点云配准，回环检测

Abstract

Design and Implementation of Multi-Sensor Simultaneous Localization and Mapping System

Currently, autonomous driving, as an intersection of computer science and automotive industry, has a broad market prospect and is rapidly becoming one of the hot areas in artificial intelligence. Autonomous driving contains key modules such as localization, perception, decision making and navigation. Simultaneous Localization and Mapping (SLAM) can effectively combine the advantages of multiple sensors to estimate the vehicle position, construct the surrounding environment, and provide a key reference for building a high-precision map containing various road condition information. In summary, accurate and reliable state estimation and map construction are important components of autonomous driving systems.

The current mainstream methods focus only on geometric feature alignment methods in static scene tasks. On the one hand, geometric feature-based alignment is very susceptible to interference from dynamic objects, which leads to accuracy degradation. On the other hand, with the development of deep semantic segmentation networks, we can easily obtain semantic information from point clouds in addition to geometric information. Semantic features can be used as a supplement to geometric features, which can improve the accuracy of odometry and loopback detection. Semantic information can also filter out dynamic objects in the point cloud data, such as pedestrians and vehicles, which can lead to residual images in the generated maps or map-based localization failures.

Through the above analysis, this paper improves the front-end odometry and loopback detection modules based on the LIO-SAM framework, and the main contributions of this paper are as follows.

(1) In this paper, we optimize the semantic information provided by the SPVNAS network and combine it with the LIDAR front-end odometer. The algorithm can still perform accurate alignment of the system in the experimental scenario with a large number of moving objects.

(2) By combining semantic information, this paper proposes a semantic-assisted Scan Context method and uses semantic ICP to verify its results and apply it as a loopback detection strategy for the system. The algorithm makes the system focus more on static scene matching during loopback detection, which can correct the cumulative error of the poses and improve the global map building effect.

(3) In this paper, we collected Jilin University campus data in various scenes using laboratory sensors, and conducted multiple correlation control experiments on the classical autopilot dataset KITTI and Jilin University campus data. The experimental results show that our method outperforms pure geometric methods, especially in dynamic scenes, and has good generalization ability.

Key Words:

Artificial intelligence, autonomous driving, semantic segmentation, simultaneous localization and mapping (SLAM), LiDAR inertial odometry, loop closure detection.

目　　录

第 1 章 绪论

1.1 研究背景与意义

近年来，随着国内人均汽车拥有量不断增加，道路拥堵情况愈发严重，交通事故发生率也屡创新高。自动驾驶作为智慧城市的重要组成部分，可以有效的提升城市运行效率和减少交通事故发生[1]。因此，全球学术界和工业界对自动驾驶的研究投入了大量的资金和精力，也取得了可观的进展。作为计算机和汽车的交叉领域，自动驾驶研究还有很大的上升空间，距离完全落地仍然任重而道远。

不同于传统地图，高精度地图可提供车道级的路网信息，在自动驾驶车辆车道级定位和导航方面有着不可替代的作用。在高精度地图使用过程中，自动驾驶车辆通过当前场景的数据，在高精度地图中进行匹配，完成车辆的定位。进而通过地图上红绿灯等语义信息，参与到车辆的导航决策过程中。因此，在自动驾驶技术中如何构建一张精度很高的地图是至关重要的。

高精度地图的制作流程包括数据采集、地图构建、自动标注、人工验证等关键步骤。而即时定位与建图（Simultaneous Localization and Mapping，SLAM）[2, 3]技术完成车辆实时位姿的估计并同步完成周围环境的建图，是地图构建步骤十分关键的环节。基于视觉的传感器，比如单目相机、双目相机、RGBD 相机等，在 SLAM 中有着广泛的应用。但是，它们十分依赖光照条件，无法胜任逆光、夜间等复杂的自动驾驶场景。相比较而言，激光雷达因其对光照不敏感等优良特性，成为了自动驾驶车辆中不可或缺的传感器。由此，激光雷达 SLAM 在工业中，有着十分广泛的应用。所以，本文采用激光雷达（Light Detection And Ranging，LiDAR）和惯性测量单元（Inertial Measurement Unit，IMU）融合的 SLAM 系统来构建高精度点云地图。

目前，绝大多数的激光雷达 SLAM 算法是针对静态场景设计的，即在搭载激光雷达车辆的运动中，周围环境自始至终均保持不变。然而，在现实生活中，这样纯静态的场景并不常见，更常见的是由人、车辆等动态物体和建筑物、植被等静态物体共同构建的复杂场景。这就对传统基于静态场景的 SLAM 方法提出了挑战。因为其采用的点云配准方式，将重投影后的距离最近的点、线、面等几

何特征形成匹配对来优化相邻帧间的位姿。而由于真实场景中运动规律不确定的移动物体的存在，在前端里程计匹配过程中会形成错误的匹配关系，利用这些匹配关系进行位姿优化，也会出现偏差。而这种偏差往往随着移动物体数量的增加而增大，最终将会影响整体的定位效果和建图质量。因此，在实际生活中更加为常见的动态场景下，解决车辆的定位与建图问题，有着更为迫切的需求和现实意义。

1.2 国内外发展现状

从上个世纪 70 年代起，随着计算机技术的快速发展，美国、英国等发达国家率先对自动驾驶技术进行了研究。2004 年，随着卡内基梅隆大学获得自动驾驶挑战赛冠军，各大高校和企业纷纷入局[4]。2009 年 4 月，Google 开启了 Waymo 自动驾驶研究计划。2018 年，Waymo 推出叫车软件，开始商业化运营。而国内的自动驾驶竞争也十分激烈，各大互联网企业，如百度、滴滴等，争相涌入，而许多自动驾驶初创公司，如 momenta、小马智行等，也来势汹汹。2020 年 10 月，百度在北京开启自动驾驶出租车服务。2021 年 12 月，北京正式开放国内首个自动驾驶出行服务商业化试点，百度和小马智行成为首批获许的企业。

在高精度地图方面，TomTom、百度等公司有着不错的表现。TomTom 是位于荷兰主打地图业务的公司，于 2017 年 7 月，与百度达成合作，共同进行高精度地图的研究工作。百度在 2017 年 4 月，开源了 Apollo 自动驾驶框架，目前已迭代至 6.0 版本，可以进行半自动化的高精度地图制作。在 2016 年 10 月，高德地图为其合作伙伴提供自动驾驶地图服务，目前已研制出第三代车载导航，并于 2022 年 1 月，确定小鹏汽车为其首家搭载企业。

SLAM，作为机器人在未知环境中进行位姿估计和地图构建的技术，在国内外也涌现了大量优秀的研究。按照所涉及的传感器大致可以分为基于相机的方案(VO)[5, 6], 基于相机和 IMU 的融合方案(VIO)[7], 纯激光雷达方案(LO)[8, 9]，激光雷达和 IMU 融合方案(LIO)[10-13]以及其他传感器方案[14]。按照所使用的后端优化方案又可以分为滤波器（Filter-Base）[10, 11]和图优化（Graph-Base）[6, 7, 9, 12-14]两种方式。SLAM 技术从上个世纪 70 年代用于火星探测开始，经历了多年的发展历程，所涉及的传感器从传统的相机再到近年来火热的激光雷达，后端优化也从卡

尔曼滤波开始，到发展了多种卡尔曼滤波器变种，再到图优化。SLAM 的实时性和准确性有很大的提高，可以处理更加复杂更大规模的室外场景。在简述完自动驾驶、高精度地图和 SLAM 的发展现状后，本文将结合所涉及的技术领域，具体地从激光雷达 SLAM、语义分割和语义 SLAM 三个方向阐述其发展现状。

1.2.1 激光雷达 SLAM 国内外发展现状

按照激光雷达线束数量不同，可以把激光雷达 SLAM 分为二维激光雷达 SLAM 和三维激光雷达 SLAM。

二维激光雷达 SLAM 常见于扫地机器人、仓库货运机器人等封闭场景中。Grisetti 等人提出的 Gmapping[15]基于粒子滤波算法，使用 RBPF 算法构建二维占据栅格地图，可以室内小场景下的实时运行。而由谷歌开发的 cartographer[16-18]使用 ceres 作为后端优化库，提出了基于 submap 的扫描匹配方法，采用分支界限法搜索回环检测，在算力有限的情况下，也可以实时完成大规模建图任务。

在三维激光雷达 SLAM 中，按照里程计中配准方式可以分为两类：基于匹配的（matching-base）和基于特征的（feature-base）。Besl 等人[19]提出的迭代最近点（Iterative Closest Point，ICP），是基于匹配里程计的基础算法。ICP 逐点迭代计算并优化帧与帧之间的误差函数，直到收敛为止。Mendes 等人[20]采用 ICP 算法构建基于姿态图的 SLAM 框架，基于 Velodyne LiDAR（HDL）传感器实现了良好的性能。而 Biber 等人[21]提出的正态分布变换（Normal Distributions Transform，NDT）是另一种基于匹配的里程计的常用方法，该方法通过体素近似的点云分布来估计姿态。Koide 等人[22]提出了一种基于 NDT 和图优化的建图方法，提供了一种可扩展的多传感器融合 SLAM 解决方案。基于匹配的里程计有很大的局限性。当点云稀疏或连续帧不扫描同一物体的同一位置时，位姿估计会有较大偏差。

除了上述基于匹配的方法，激光雷达里程计与建图（Lidar Odometry and Mapping，LOAM）[8]方法中使用的基于特征的方法已经成为基于雷达的 SLAM 方案中较为流行的前端解决方案。LOAM 提出了使用计算局部区域的曲率信息提取边缘特征和平面特征的方法。而 LeGO-LOAM[13]在 LOAM 的基础上增加了宽度优先搜索（Breadth-First Search，BFS）聚类方法，以便过滤数据中一些离散点。虽然一些干扰特征被过滤掉了，但是一些关键特征也因为上述方法被过滤从

而造成信息丢失。本文基于 LIO-SAM[12]进行改进，LIO-SAM 是一个比较优秀的特征匹配 SLAM 框架。它的主要思想是基于因子图将姿态估计问题转化为最大后验问题，进而使用图优化的方法求解最优位姿。尽管这些方法在某些实验条件下具有出色的建图精度，但其泛用性、鲁棒性并不好，尤其是在动态对象较多的场景中。动态物体的存在增加了里程计配准的不确定性，导致姿态估计精度下降。同时，生成地图中也会残留动态物体的运动轨迹，这给后续回环检测和重定位问题带来了挑战。

除了前端里程计，回环检测问题也是构建高精度地图过程中的关键。回环检测的本质是场景重识别问题，它通过计算当前帧与历史帧的相似度来判断是否重新访问了相同的地方。如果存在回环，则会在优化过程中加入新的约束，以消除帧间累积计算带来的误差[23]。目前回环检测方法可以分为两类：局部描述符和全局描述符[24]。局部描述符方法的主要思想是通过构建词袋（Bag of words，BoW）[25]模型生成数据的局部特征描述构建词库来计算场景之间的相似度。Steder 等人[26]提出了一种使用 BoW 和特征点信息相结合的回环检测方法，这是一种典型的将局部描述符应用于 SLAM 框架的方法。然而，这些特征提取的方法需要提前构建词库，因而不能用于实时的回环检测过程。此外，这些局部描述符表达能力有限，不足以区分高度相似的场景。

与局部描述符不同，全局描述符不需要提前构建词库。他们多采用点与点之间的几何关系编码生成直方图或矩阵，以计算帧之间的相似度[24, 27, 28]。形状函数集合(Ensemble of Shape Functions，ESF)[28]和视点特征直方图(Viewpoint Feature Histogram，VFH)[27]通过提取以直方图形式表示的全局描述符，提高了算法的匹配能力和鲁棒性。而 Scan Context[24]方法通过 bin 编码函数将 3D 点云映射为矩阵，并使用两步搜索策略提高其效率。与局部描述符相比，虽然前面提到的全局描述符在一定程度上提高了算法的鲁棒性，但判别问题仍然存在。为了提高全局描述符的可区分性，可考虑在描述符编码过程中引入语义信息。

1.2.2 语义分割国内外发展现状

环境感知是自动驾驶中十分重要的部分，而目标检测和语义分割是自动驾驶环境感知中不可或缺的算法。目标检测是利用图像或者点云获取车辆、行人、障

碍物等物体的空间位置、类别和行驶方向等关键信息，而语义分割是利用图像或者点云获取每个像素点或者点云点的类别信息。点云相比于图片，信息量更大和维度更高，是语义分割研究中十分热门的话题。

传统的点云语义分割方法是利用经典机器学习模型，先利用聚类算法将点云分割为若干个相对独立的小物体，然后使用支持向量机（Support Vector Machine，SVM）[29]、自适应提升算法（Adaptive Boosting，AdaBoost）[30]等进行分类和预测。这种特征提取和分类更关注小范围的点云关联信息，从而忽略了大范围的上下文关系，而这种关系在语义分割中是十分重要的。目前主流的语义分割方法是神经网络，其按照处理点云的组织形式可以分为基于点（Point-Base）、基于投影（Projection-Base）、基于体素（Voxel-Base）等方式。

基于点（Point-Base）：比较经典的是 PointNet[31]和 PointNet++[32]。PointNet 通过 T-Net 学习相对变换，通过共享多层感知机（Multilayer Perception，MLP）来进行局部特征提取，通过最大池化层将其进一步提取为全局特征向量，该向量可以直接进行全连接层分类。而对于语义分割而言，需要将局部特征向量和全局特征向量进行合并，再使用 MLP 对每个点完成标签分配。而 PointNet++在 PointNet 的基础上，增加了对邻域信息的利用，设计了多层级聚类采样的方式（Set Abstraction，SA）。这样得到的局部特征具有更大的感受野，可以整合更广范围的信息。再结合下采样层和上采样层之间的跳线（skip link）连接，对前后的特征层完成整合，从而更准确的预测每个点的语义标签。

基于投影（Projection-Base）：点云处于三维空间中，而在三维空间中进行三维卷积的操作，时间和空间的复杂度都是高昂的，所以一些方法会将点云投影至二维平面上，再进行相关运算。Mulioto 等人的 RangeNet++[33]，会先将点云投影至 RangeView 上，然后输入到神经网络中得到每个点类别，再使用膨胀和腐蚀减少类别出错的点，最后反投影回三维空间，这种操作实质上提高了语义结果的局部一致性。又比如 Felix Jaremo 等人[34]将点云投影成一系列的二维图片，然后使用二维卷积神经网络处理这些图片，最后将多张图片的预测结果融合投影到点云中。

基于体素（Voxel-Base）：基于体素的方法实质是将大量的点云划分到有限的体素中，进行结构化的表示。而常见的做法为将点云分为若干个体素，然后直接

放入到全 3D 卷积神经网络（Fully-3D CNN）中，结果很容易受体素边界伪影的影响。为解决这一问题，SegCloud[35]引入了三线性插值处理栅格的边界情况，再用 3D CNN 预测出粗略的语义标签，最后使用全连接条件随机场（Fully Connected Conditional Random Field）对标签进行处理，从而使预测出的语义标签具有空间上的一致性。

1.2.3 语义 SLAM 国内外发展现状

深度学习在图像和点云数据处理上有着广泛的应用，比如目标检测、语义分割等等。许多研究人员开始尝试在 SLAM 系统中加入语义信息，进而在已有的约束条件中引入语义约束，从而提高位姿估计的准确性。Renato F 等人[36]提出的 SLAM++，在提供物体的几何先验模型的情况下，可以利用深度图片对静态场景中的已知物体位姿进行预测，再与相机间 ICP 得到的相对位姿，共同构建因子图并进行优化。因此，该算法可以在大型杂乱的场景中，进行实时定位和物体级场景的生成。Sean L. Bowman 等人[37]将几何信息和语义信息放入一个紧耦合的优化框架中，并将语义 SLAM 的联合优化问题解耦为两个子问题：连续的位姿求解问题和离散的数据关联问题。在不同光照条件下的室内外杂乱场景中有着较好的表现。Parv Parkhiya 等人[38]首先标注并训练大量 CAD 模型的关键点，并将不同帧间的特征点进行数据关联，最后将其与视觉里程计因子融合组成因子图，进行优化求解。Chao Yu 等人[39]提出的 DS-SLAM，利用图像语义分割得到每个像素点的类别信息，进而结合语义一致性检查、过滤动态物体，随后使用单独线程进行建图工作，最终生成语义稠密的 3D 八叉树地图。Xieyuanli Chen 等人[9]提出的 SUMA++，将语义信息集成到面元表示的地图中，并且使用类别标签过滤动态物体。该算法在 KITTI 数据集中，相比于单纯依靠几何面元的方法和单纯使用标签过滤动态物体的方法，拥有更好的建图效果。

1.3 本文研究内容和章节安排

1.3.1 本文的研究内容

本文针对大规模室外动态场景的建图问题，分析了现有算法存在的问题，基

于 LIO-SAM 框架，提出了语义信息参与的 SLAM 系统。

（1）针对动态场景中的点云配准问题，本文提出了一种将语义信息融入到激光雷达里程计的算法。通过语义分割网络获得的语义信息，经过纠正模块和过滤动态物体后，用以参与点云配准，为其提供额外的语义约束，减少误匹配的产生，增加点云配准的准确率。具体实现在 3.3 节详细展开说明。

（2）针对动态场景中的回环检测问题，本文提出了一种将语义信息融入到回环检测的算法。通过构建带有语义的 Scan Context，借鉴注意力机制，使得其在考虑帧与帧间的相似度时，更加关注静态固定的物体，增加了回环检测的准确性和鲁棒性。具体实现在 3.4 节详细展开说明。

（3）针对改进方案的效果验证问题，本文在公开数据集 KITTI 以及使用实验室采集平台采集的吉林大学校园数据上进行了多方面验证，详细实验设置及内容参见 4.2 节。其结果表明，本文的方法在动态场景中，相比 Lego-LOAM 和 LIO-SAM 有着更佳的表现。

1.3.2 本文的章节安排

第 1 章，绪论。本章针对课题的研究背景和现实意义进行了说明和解析，针对激光雷达 SLAM、语义分割和语义 SLAM 的国内外研究现状进行了详细的总结，最后对本论文的整体结构和章节安排进行概述。

第 2 章，基础理论及相关软硬件。本章对论文坐标和运算符号进行统一定义，对旋转表示及其相关公式进行简要介绍，由非线性优化到图优化的相关公式运算进行了说明，对本文中使用的传感器工作原理及其相关算法进行介绍，最后，简要介绍了系统实现中使用的软件库。

第 3 章，多传感器融合的 SLAM 设计与实现。本章详细介绍了多传感器融合的 SLAM 的整体设计。首先，给出了整体的因子图结构，紧接着对其中所涉及的 IMU 约束、LiDAR 约束和回环检测约束。其中，重点介绍了语义信息在点云配准中的语义约束作用，为其实验结果提供理论基础。

第 4 章，实验结果与分析。本章首先对实验数据采集平台的各种传感器进行展示和介绍，然后简要列举算法实验中使用的数据集，KITTI 数据集和我们采集的吉林大学数据的相关概要，之后进行了多组相关实验，对我们算法和其他优秀算法在数据集上的表现进行了分析和总结。

第 5 章，总结与展望，本章充分探讨了算法的优点以及存在不足，并对本研和实验中存在的不足进行反思，同时对未来算法的改进方向提出了若干意见。

第 2 章 基础理论知识及相关软硬件

2.0 坐标系定义

在本文中，所涉及的坐标系有世界坐标系（World）、IMU 坐标系、激光雷达坐标系。其中，本文选用 IMU 坐标系作为载体坐标系（Body），它们之间的关系如下图所示：

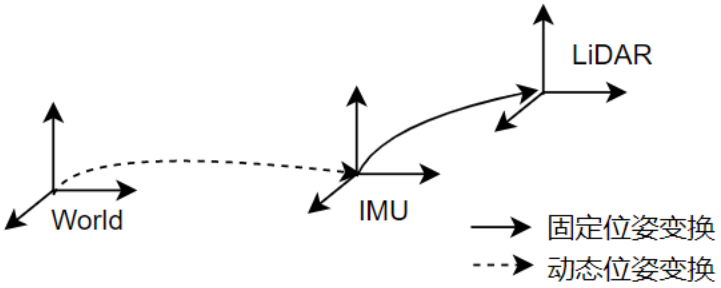

图 2.1 坐标系示意图

2.1 符号说明

本文中括号右上角标注的单一英文字母表示其所属坐标系信息。如，$(\cdot)^{W}$表示世界坐标系，$(\cdot)^{B_k}$表示第 k 帧的 IMU 坐标系，同理，$(\cdot)^{L_k}$表示第 k 帧的 LiDAR 坐标系。

括号右上及右下标注字母表示变换关系，$(\cdot)_{B}^{A}$表示从参考系 B 到参考系 A 的变换关系。如，R_B^A表示从参考系 B 到参考系 A 的旋转矩阵，P_B^A表示从参考系 B 到参考系 A 的空间变换，同理，T_B^A表示从参考系 A 到参考 B 的位姿变换矩阵。

$\widehat{(\cdot)}$ 表示该变量为带有噪声项的测量值或者估计值。

本文中，载体的旋转表示方式有旋转矩阵、旋转向量、欧拉角、四元数四种方式，其定义和符号说明如下：

（1）旋转矩阵

旋转矩阵的定义为$R \in \mathbb{R}^{3\times3}$，且满足单位正交，即$RR^T = \mathrm{I}$ 、$\det(R) = 1$。显然，旋转矩阵的逆（转置）表示一个相反方向的旋转。$R \in \mathrm{SO}(3)$, $\mathrm{SO}(3)$为欧式特殊正交群。

（2）旋转向量（轴角）：

使用旋转轴$\mathbf{n}$和旋转角ϕ来表示旋转。

$$\boldsymbol{\Theta} = \mathbf{n}\phi = \begin{bmatrix} \mathrm{n}_x\phi \\ n_y\phi \\ n_z\phi \end{bmatrix} = \begin{bmatrix} \Theta_x \\ \Theta_y \\ \Theta_z \end{bmatrix} \cdots\cdots (2\text{-}1)$$

其中，旋转轴$\mathbf{n}$需为单位向量，即$\|\mathbf{n}\| = 1$。

（3）欧拉角：

欧拉角是一种使用三个绕坐标轴的旋转角来表示旋转的方式，Roll 表示绕 X 轴的滚转角，Pitch 表示绕 Y 轴的俯仰角以及 Yaw 表示绕 Z 轴的偏航角。需要注意的是，即使相同的三个角，不同的旋转顺序也会导致不同的旋转结果。本文采用的顺规为 ZYX，也称为 YPR 顺规，即先绕 Z 轴旋转、再绕 Y 轴旋转、最后绕 X 轴旋转得到的旋转结果。

YPR 顺规欧拉角到旋转矩阵变换可以表示为：

$$\begin{aligned} \mathrm{R}\left\{\begin{bmatrix} yaw \\ pitch \\ roll \end{bmatrix}\right\} &= \mathrm{R}\left\{\begin{bmatrix} 0 \\ 0 \\ yaw \end{bmatrix}\right\} \times \mathrm{R}\left\{\begin{bmatrix} 0 \\ pitch \\ 0 \end{bmatrix}\right\} \times \mathrm{R}\left\{\begin{bmatrix} roll \\ 0 \\ 0 \end{bmatrix}\right\} \\ &= \begin{bmatrix} c_1c_2 & c_1s_2s_3 - c_3s_1 & s_1s_3 + c_1c_3s_2 \\ c_2s_1 & c_1c_3 + s_1s_2s_3 & c_3s_1s_2 - c_1s_3 \\ -s_2 & c_2s_3 & c_2c_3 \end{bmatrix} \cdots\cdots (2\text{-}2) \end{aligned}$$

其中，

$$c_1 = \cos(\mathrm{yaw}), c_2 = \cos(\mathrm{pitch}), c_3 = \cos(\mathrm{roll}),$$

$$s_1 = \sin(\mathrm{yaw}), s_2 = \sin(\mathrm{pitch}), s_3 = \sin(\mathrm{roll}).$$

（4）四元数：

四元数通过一个实部和三个虚部来表示旋转，如下式。

$$\mathbf{q} = [q_w, \boldsymbol{q}_v]^T,\ q_w \in \mathbb{R},\ \boldsymbol{q}_v = \left[q_x, q_y, q_z\right]^T \in \mathbb{R}. \cdots\cdots (2\text{-}3)$$

而单位四元数，即$\|\mathbf{q}\| = 1$，可以表示一个旋转。后文如无特殊说明，四元数和单元四元数含义一致。四元数的乘法计算过程如下式所示：

$$\mathrm{p}\otimes\mathrm{q} = \begin{bmatrix} p_wq_w - \boldsymbol{p}_v^T\boldsymbol{q}_v \\ p_w\boldsymbol{q}_v + q_w\boldsymbol{p}_v + \boldsymbol{p}_v \times \boldsymbol{q}_v \end{bmatrix} \cdots\cdots (2\text{-}4)$$

将坐标系 A 中的坐标点转换到坐标系 B 中，其计算过程为：

$$P^A = \mathbf{q}_B^A \otimes P^B \mathbf{q}_B^{A*} = R\{\mathbf{q}_B^A\}P^B \cdots\cdots (2\text{-}5)$$

其中，$\mathbf{q}^*$表示为$\mathbf{q}$的共轭四元数。$R\{\mathbf{q}_B^A\}$表示将$\mathbf{q}_B^A$转换成的旋转矩阵。

2.2 旋转群 SO(3)及其李代数

由所有旋转矩阵组成的集合叫做特殊正交群$SO(3)$，与$SO(3)$对应的李代数为$\mathfrak{so}(3)$，其包含的元素为反对称矩阵[40]。设$\omega = [\omega_x, \omega_y, \omega_z]$，则有运算符$(\cdot)^\wedge$将三维向量映射为反对称矩阵：

$$\omega^\wedge = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \cdots\cdots (2\text{-}6)$$

而运算符$(\cdot)^\vee$则将反对称矩阵映射为三维向量：

$$\begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}^\vee = \omega \cdots\cdots (2\text{-}7)$$

指数映射与对数映射，其公式如下式所示：

$$\mathrm{R} = \mathrm{Exp}(\Theta) = \exp(\Theta^\wedge) \cdots\cdots (2\text{-}8)$$

$$\Theta = \mathrm{Log}(\mathrm{R}) = \log(\mathrm{R})^\vee \cdots\cdots (2\text{-}9)$$

BCH 公式：

关于两个李代数指数映射之间的乘积的对数映射，可以由 Baker–Campbell–Hausdorff（BCH）公式[41]进行展开：

$$\ln(\exp(\mathrm{A})\exp(\mathrm{B})) = A + B + \frac{1}{2}[A,B] + \frac{1}{12}\big[A,[A,B]\big] - \frac{1}{12}\big[B,[A,B]\big] + \cdots\cdots$$

$$\cdots\cdots (2\text{-}10)$$

其中，[·]表示李括号运算。

当其中一个李代数为小量时，可以进行以下的近似展开：

$$\ln\left(\exp\left(\phi_1^\wedge\right)\exp\left(\phi_2^\wedge\right)\right)^\vee \approx \begin{cases} J_l(\phi_2)^{-1}\phi_1 + \phi_2, & if\ \phi_1 is\ small \\ J_r(\phi_1)^{-1}\phi_2 + \phi_1, & if\ \phi_1 is\ small \end{cases} \cdots\cdots (2\text{-}11)$$

其中，

$$J_l = \frac{\sin\theta}{\theta} I + \left(1 - \frac{\sin\theta}{\theta}\right) aa^T + \frac{1-\cos\theta}{\theta} a^\wedge$$

$$J_l^{-1} = \frac{\theta}{2}\cot\frac{\theta}{2} I + \left(1 - \frac{\theta}{2}\cot\frac{\theta}{2}\right) aa^T - \frac{\theta}{2} a^\wedge$$

并有如下性质：

$$J_r(\phi) = J_l(-\phi)$$

扰动模型：

在 SLAM 的优化问题中，时常需要求解对旋转或者位姿的导数。将旋转或者位姿使用李群表示后，左乘或者右乘微小扰动，再对该扰动求导的方式，称为扰动模型。以下给出关于旋转和位姿的扰动模型[23]。

SO（3）：考虑旋转矩阵的情况，对于一个空间点P进行旋转变换 R（对应的李代数为ϕ），得到$P' = RP$，则旋转后的点相对于旋转矩阵的导数为：

$$\frac{\partial(RP)}{\partial\varphi} = -(RP)^{\wedge}. \cdots\cdots(2\text{-}12)$$

SE（3）：考虑位姿变换矩阵的情况，对于一个空间点 P 进行位姿变换 T（对应的李代数为），得到$P' = TP$，则变换的点相对李代数的导数为：

$$\frac{\partial(TP)}{\partial\delta\xi} = \begin{bmatrix} I & -(RP+t)^{\wedge} \\ \mathbf{0}^T & 0 \end{bmatrix} = (TP)^{\odot}. \cdots\cdots(2\text{-}13)$$

2.3 四元数相关运算

（1）四元数相对于向量求导，其计算过程如下：

$$\frac{\partial(\mathbf{q}\otimes\mathbf{a}\otimes\mathbf{q}^*)}{\partial\mathbf{a}} = \frac{\partial(\mathbf{R}\{\mathbf{q}\}}{\partial\mathbf{a}} = \mathbf{R}\{\mathbf{q}\} \quad \cdots\cdots(2\text{-}14)$$

（2）四元数相对于四元数求导，其计算过程如下：

$$\frac{\partial(\mathbf{q}\otimes\mathbf{a}\otimes\mathbf{q}^*)}{\partial\mathbf{q}} = 2[q_w\mathbf{a} + \mathbf{q}_v \times \mathbf{a} \mid \mathbf{q}_v{}^T\mathbf{a}\mathbf{I}_{3\times3} + \mathbf{q}_v\mathbf{a}^T - q_w\mathbf{a}^{\wedge}] \cdots\cdots(2\text{-}15)$$

（3）四元数相对与旋转增量的求导，其计算过程如下：

$$\frac{\partial(\mathbf{q}\otimes\mathbf{a}\otimes\mathbf{q}^*)}{\partial\delta\boldsymbol{\theta}} = \frac{\partial(\mathbf{R}\{\mathbf{q}\}\mathbf{a})}{\partial\delta\boldsymbol{\theta}} = -\mathbf{R}\{\boldsymbol{\theta}\}\mathbf{a}^{\wedge}\mathbf{J}_r(\boldsymbol{\theta}) \cdots\cdots(2\text{-}16)$$

2.4 非线性优化

SLAM 中的优化问题，最终往往可以表示为非线性最小二乘问题。而求解 SLAM 优化问题也就变成了求解非线性最小二乘的最优值问题。由于转化后的最小二乘问题十分复杂，无法直接求解极小值，所以比较常用的方法是使用迭代法。本文以典型的最小二乘法问题[42]为例：

$$\min_x F(x) = \frac{1}{2}\|f(x)\|_2^2 \cdots\cdots(2\text{-}17)$$

迭代的步骤为：

1. 给定一个初始值x_o。
2. 在第 K 次迭代的时候，寻找到增量Δx_k，使得$\|f(x_k + \Delta x_k)\|_2^2$取到极小值。

3. 如果Δx_k或者 $f(x_k+\Delta x_k)-f(x_{k-1}+\Delta x_{k-1})$ 足够小的时，则停止。
4. 否则，令$x_{k+1}=x_k+\Delta x_k$，再进行第二步。

这样，只要不断找到合适的增量，便会不断逼近极小值。常见的迭代法有高斯牛顿法[43]和列文伯格-马夸尔特法[44]：

高斯牛顿法：增量方程为：

$$\mathrm{J(x)J^T}(x)\Delta x=-J(x)f(x)\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(2\text{-}18)$$

算法步骤为：

1. 给定一个初始值x_o。
2. 在第 K 次迭代的时候，计算出雅克比矩阵 $\mathrm{J}(x_k)$和误差$\mathrm{f}(x_k)$。
3. 在增量方程求解Δx。
4. 否则，令$x_{k+1}=x_k+\Delta x_k$，再进行第二步。

列文伯格-马夸尔特法：增量方程为：

$$(\mathrm{J(x)J^T}(x)+\lambda I)\Delta x=-J(x)f(x)\cdots\cdots\cdots\cdots\cdots\cdots(2\text{-}19)$$

算法步骤为：

1. 给定一个初始值x_o以及初始半径μ。
2. 在第 K 次迭代的时候，计算出雅克比矩阵 $\mathrm{J}(x_k)$和误差$\mathrm{f}(x_k)$。
3. 在增量方程求解Δx_k。
4. 计算$\rho=\frac{f(x_k+\Delta x_k)-f(x_k)}{\mathrm{J}(x_k)^T\Delta x_k}$。
5. 若$\rho>\frac{3}{4}$，则设置$\mu=2\mu$。
6. 若$\rho<\frac{1}{4}$，则设置$\mu=2=0.5\,\mu$。
7. 如果ρ大于某种阈值，令$x_{k+1}=x_k+\Delta x_k$，再进行第二步；否则，则终止算法。

上述两种优化方法，可以使用 G2O、Ceres 或者 GTSAM 等开源库构建问题，调用对应的优化方法，得到最终优化量。

2.5 位姿 SLAM 与图优化

2.5.1 位姿图

经典的图优化方案，是将机器人的位姿和地图路标（Landmark）的 3 维坐标作为待优化参数，进行整体的优化，最终得到机器人轨迹和周围环境地图，如图 2.2 左侧所示。实际上，大多数的路标点在优化几次后，便会收敛到一个位置，所以可以构建只包含机器人位姿的位姿图（Pose Graph）如图 2.2 右侧所示。在位姿图中，节点表示待优化的位姿，而节点与节点之间的边使用特征匹配的方式获得初始值，如图 2.2。这种只有优化位姿的方式，可以减少计算量，提高后端优化的实时性。

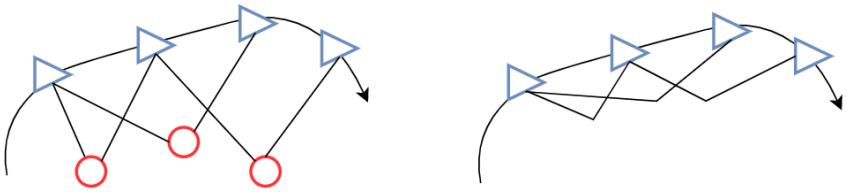

图 2.2 普通图优化和位姿图优化对比示意图

于是，我们得到只优化位姿的因子图（factor graph）优化方案[45]。因子图的点集合为$\{x_i\}$，一元边集合$\{e_i\}$，二元边集合$\{e_{ij}\}$。于是，因子图整体的损失函数为：

$$F(X) = \sum_i e_i^T \Omega_i e_i + \sum_i e_{ij}^T \Omega_i e_{ij} \quad \cdots\cdots (2\text{-}20)$$

其中，e_i、e_{ij}分别表示一元边、二元边对应的残差项，Ω_i表示一元边或者二元边对应的噪声模型或者协方差。

整体的优化目标为：

$$X^* = \underset{X}{\operatorname{argmin}} F(X) \quad \cdots\cdots (2\text{-}21)$$

2.5.2 一元边的位姿修正

一元边，即在因子图中仅和一个节点相关联的边，表示对某个节点位姿的直

接观测。在实际工程中，这种一元边往往来自于 GPS 的观测数据，又或者定位中当前帧与地图的匹配结果。如何调整节点位姿，使其更接近观测结果，是待解决的问题。

首先，构建节点位姿和观测数据的残差项，其计算方法如下式所示：

$$e_i = \ln(Z_i^{-1}T_i)^\vee = \ln(\exp((-\xi_{zi})^\wedge)\exp(\xi_i^\wedge)) \cdots\cdots\cdots\cdots (2\text{-}22)$$

其中，T_i表示第i个节点待优化位姿，Z_i表示第i个节点的观测数据，ξ_{zi}、ξ_i分别表示待优化位姿和观测数据对应的李代数。理想情况下，e_i应该为 0，但是由于观测噪声的存在，e_i大于 0，于是通过调整位姿，使得e_i尽可能的小。

然后，在残差项中添加扰动，用以计算对应的雅克比矩阵。

$$\hat{e}_i = \ln\left(Z_i^{-1}exp(\delta\xi_i^\wedge)T_i\right)^\vee \cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}23)$$

使用 BCH 公式，并运用李代数的伴随性质，可以化简为：

$$\hat{e}_i = e_i + J_r^{-1}(e_i)Ad(T_i^{-1})\ \delta\xi_i \cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}24)$$

于是，可以得到残差相对于位姿扰动的雅克比：

$$\frac{\partial \hat{e}_i}{\partial \delta\xi_i} = J_r^{-1}(e_i)Ad(T_i^{-1}) \cdots\cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}25)$$

其中，

$$J_r^{-1}(e_i) \approx I + \frac{1}{2}\begin{bmatrix}\phi_e^{\mathrm{T}} & \rho_e^{\mathrm{T}} \\ \mathbf{0}^T & \phi_e^{\mathrm{T}}\end{bmatrix}$$

2.5.3 二元边的位姿修正

二元边，即在因子图中和两个节点都有关联的边，表示对某两个节点相对位姿的观测。在实际工程中，这种二元边往往来自于相邻帧的相对位姿估计和回环检测。其中，相邻帧的相对位姿估计是和传感器相关的，比如 IMU 预积分、相机的 BA（Bundle Adjustment，光束平差法）和激光雷达的点云配准算法。如何调整两个节点位姿，使其更接近观测结果，是优化中要解决的问题。

首先，构建节点位姿和观测数据的残差项：

$$e_{ij} = \ln\left(Z_{ij}^{-1}T_i^{-1}T_j\right)^\vee = \ln(\exp((-\xi_{ij})^\wedge)\exp((-\xi_i)^\wedge)\exp(\xi_j^\wedge)) \cdots (2\text{-}26)$$

其中，T_i、T_j分别表示第i、j个节点的待优化位姿，Z_{ij}表示第i、j个节点相对位姿

的观测数据，ξ_i、ξ_j、ξ_{ij}分别表示待优化位姿和观测数据对应的李代数。理想情况下，e_{ij}应该为 0，但是由于观测噪声的存在，e_{ij}大于 0，于是调整位姿的情况，使得e_{ij}尽可能的小。

然后，在残差项中对节点位姿添加扰动，用以计算其对应的雅克比矩阵。

$$\hat{e}_{ij} = \ln\left(Z_{ij}^{-1}T_i^{-1}exp((-\delta\xi_j)^\wedge)exp(\delta\xi_j{}^\wedge)T_j\right)^\vee \cdots\cdots\cdots\cdots(2\text{-}27)$$

使用 BCH 公式，并运用李代数的伴随性质，可以化简为：

$$\hat{e}_{ij} = e_{ij} - J_r^{-1}(e_{ij})Ad(T_j^{-1})\delta\xi_i + J_r^{-1}(e_{ij})Ad(T_j^{-1})\delta\xi_j \cdots\cdots(2\text{-}28)$$

于是，可以得到残差相对于位姿扰动的雅克比：

$$\frac{\partial\hat{e}_{ij}}{\partial\delta\xi_i} = -J_r^{-1}(e_{ij})Ad(T_j^{-1}) \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(2\text{-}29)$$

$$\frac{\partial\hat{e}_{ij}}{\partial\delta\xi_j} = J_r^{-1}(e_{ij})Ad(T_j^{-1}) \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(2\text{-}30)$$

其中，

$$J_r^{-1}(e_{ij}) \approx I + \frac{1}{2}\begin{bmatrix}\phi_e^{\mathrm{T}} & \rho_e^{\mathrm{T}} \\ \mathbf{0}^T & \phi_e^{\mathrm{T}}\end{bmatrix}$$

2.5.4 马氏距离

上式中，残差的马氏距离[46]需要转换为 2 范数的最小二乘问题，才能继续套用之前的非线性优化手段，所以马氏距离可以用下式完成转换:

$$\|e\|_\Sigma^2 = e^T\Sigma^{-1}e = (\Sigma^{-1/2}e)^T(\Sigma^{-1/2}e) = \left\|\Sigma^{-1/2}e\right\|_2^2 \cdots\cdots\cdots(2\text{-}31)$$

2.6 传感器测量模型及相关算法

2.6.1 LiDAR

激光雷达（Light Detection And Ranging，简称为 LiDAR），通过一束激光脉冲测量周围环境的距离等参数。根据发射方式的不同可以分为固态激光雷达和机械式激光雷达，前者是朝着固定的方向，后者可以绕轴进行 360 度的旋转。一个典型的机械式激光雷达是由激光发射器、斜面镜、负责旋转的电机、激光接收器等

部分组成。相比于相机而言，激光雷达不易受光线变化影响，可以直接提供三维深度信息，测量精度高。近年来，激光雷达也广泛的应用于自动驾驶领域。本次实验采用的是 Velodyne HDL-32E 激光雷达，它是由激光雷达顶级厂商 Velodyne 生产的一款 32 线的激光雷达。

对于一根激光线来讲，从发射器出发、接触不可穿透物体返回、接收器收到，可以通过计算飞行时间（TOF）来估计物体的距离 R，从而可以用以下公式计算该物体在激光雷达坐标系下的三维坐标：

$$\begin{cases} x = R * \cos(\omega) * \sin(\phi) \\ y = R * \cos(\omega) * \cos(\phi) \\ z = R * \sin(\omega) \end{cases} \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}32)$$

其中，ω是发射激光线的垂直角，即与 XOY 平面的夹角；ω是发射激光线的水平角，即与 YOZ 平面的夹角。

2.6.2 IMU

惯性测量单元（Inertial Measurement Unit，IMU），内部含有一个加速度计和陀螺仪，可以提供测量物体的加速度和角速度，用于估计测量物体的位姿。

（1）加速度计测量模型[47]：

$$\widehat{a_k} = a_k + b_{a,k} + R_W^{B_k} g^w + \eta_a \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}33)$$

其中，$\widehat{a_k}$为加速度计在 IMU 坐标系第 k 帧时带有噪声的测量值，a_k为第 k 帧时加速度计的真实值，$b_{a,k}$为第 k 帧加速度计对应的零偏，$R_W^{B_k}$为从世界坐标系到 IMU 坐标系的旋转矩阵，g^w为世界坐标系下的重力加速度，η_a为加速度计测量时的高斯白噪声，其满足以下分布：

$$\eta_a \sim \mathcal{N}(0, \sigma_a^2) \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}34)$$

而假定加速度零偏符合随机游走，其对时间的导数满足马尔科夫性，即：

$$\dot{b_a} = \eta_{ba} \ ，\eta_{ba} \sim \mathcal{N}(0, \sigma_{ba}^2) \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}35)$$

（2）陀螺仪测量模型[47]：

$$\widehat{\omega_k} = \omega_k + b_{g,k} + \eta_g \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}36)$$

其中，$\widehat{\omega_k}$为陀螺仪在 IMU 坐标系第 k 帧时带有噪声的测量值，ω_k第 k 帧时陀螺仪的真实值，$b_{g,k}$为第 k 帧陀螺仪计对应的零偏，η_g为陀螺仪测量时的高斯白噪

声，其满足以下分布：

$$\eta_g \sim \mathcal{N}(0, \sigma_g^2) \cdots\cdots (2\text{-}37)$$

而假定陀螺仪零偏符合随机游走，其对时间的导数满足马尔科夫性，即：

$$\dot{b_g} = \eta_{bg} \ , \eta_{bg} \sim \mathcal{N}(0, \sigma_{bg}^2) \cdots\cdots (2\text{-}38)$$

2.6.3 LiDAR 和 IMU 的标定

LiDAR 和 IMU 之间的标定十分重要，偏差很小的角度，系统的精度就会变得很低。设 LiDAR 帧间通过点云配准算法可以得到相对位姿矩阵$\mathrm{T}_{l_{k+1}}^{l_k}$，而 IMU 通过预积分可以得到相对位姿矩阵为$\mathrm{T}_{b_{k+1}}^{b_k}$，LiDAR 和 IMU 之间的外参矩阵为$\mathrm{T}_l^b$，若各个位姿计算准确的情况下，对于任一帧，应有：

$$\mathrm{T}_{b_{k+1}}^{b_k} T_l^b = T_l^b \mathrm{T}_{l_{k+1}}^{l_k} \cdots\cdots (2\text{-}39)$$

但是，由于 LiDAR 和 IMU 之间外参矩阵估计存在误差，可以构建残差项：

$$\mathrm{e} = \left(\mathrm{T}_{b_{k+1}}^{b_k} T_l^b\right)^{-1} T_l^b \mathrm{T}_{l_{k+1}}^{l_k} \cdots\cdots (2\text{-}40)$$

通过非线性优化，可以将外参矩阵优化到一个较好的解。

在实际的实验中，需要采集一小段包含 LiDAR 和 IMU 的数据构建非线性优化的最小二乘方程。另外，需要使用直尺测量 LiDAR 和 IMU 之间的大致位置关系，用作非线性优化时的初始值。

2.7 点云相关算法

2.7.1 点云配准算法

点云是一系列三维坐标点的集合，可以是通过激光雷达或者其他方式获得。而点云配准则是计算两帧之间的位姿变化，使得通过变换后两帧之间尽可能的近似。使用数学语言可以描述为：$\mathrm{P} = \{P_1, P_2, \dots, P_n\}$，$\mathrm{Q} = \{Q_1, Q_2, \dots, Q_n\}$。而点云配准就是寻找位姿变换矩阵$\mathrm{R}, \mathrm{t}$，使得$\forall \mathrm{i}, P_i = \mathrm{R}Q_i + t$。

常用的点云配准算法有迭代最近点（Iterative Closest Point，ICP）[19]、正态分布变换（Normal Distributions Transform，NDT）[21]和 Loam-Based[8]等。

ICP 算法：定义两帧点云之间的残差项为：$e_i = P_i - (\mathrm{R}Q_i + t)$，进而可以构建关于残项的最小二乘问题。

$$\min_{R,t} \frac{1}{2}\sum_{i=1}^{n} \|P_i - (\mathrm{R}Q_i + t)\|_2^2. \cdots\cdots(2\text{-}41)$$

这个最小二乘问题，可以使用 SVD 求解和非线性优化两种方法进行求解。

（1）使用 SVD 方法进行求解：通过对最小二乘问题的化简和不等式放缩可以得到极小值的正定解。算法过程可以描述为：

1. 计算两帧点云的质心$\overline{\mathrm{P}}$、$\overline{\mathrm{Q}}$，并进行去质心变换：

$$P_i' = P_i - \overline{\mathrm{P}}, Q_i' = Q_i - \overline{\mathrm{Q}}. \cdots\cdots(2\text{-}42)$$

2. 构建矩阵：

$$\mathrm{W} = \sum_{i=1}^{n} P_i' Q_i'^T. \cdots\cdots(2\text{-}43)$$

3. 对矩阵使用 SVD 分解：

$$\mathrm{W} = \mathrm{U}\Sigma V^T \cdots\cdots(2\text{-}44)$$

其中，Σ是对角线元素从大到小排列的对角矩阵，而U、V^T也为对角矩阵。

当 W 满秩时，R 为：

$$\mathrm{R} = \mathrm{U}V^T. \cdots\cdots(2\text{-}45)$$

4. 计算平移向量t：

$$\mathrm{t} = \mathrm{P} - \mathrm{RQ}.$$

（2）使用非线性优化求解：可以使用李代数重新表达最小二乘问题：

$$\min_{\xi} \frac{1}{2}\sum_{i=1}^{n} \left\|P_i - \exp(\xi^\wedge) Q_i\right\|_2^2 \cdots\cdots(2\text{-}46)$$

误差项相对于位姿的雅克比矩阵：

$$\frac{\partial e}{\partial \delta\xi} = -(\exp(\xi^\wedge) Q_i)^\odot. \cdots\cdots(2\text{-}47)$$

可以使用高斯牛顿法或 LM 法对最小二乘法进行求解。

NDT：NDT 先将点云栅格化后使用高斯分布描述，如果两帧点云位姿变化后接近，则两帧点云的栅格高斯分布比较接近，一帧点云在另一点云中的概率密度将会很大。所以可以使用求解概率密度之和最大对应的位姿，来完成点云配准。

1. 将点云 P 进行栅格化处理。
2. 计算每个格子的高斯分布参数：

$$\mu = \frac{1}{n}\sum_{i=1}^{n} P_i \cdots\cdots(2\text{-}48)$$

$$\Sigma = \frac{1}{n-1}\sum_{i=1}^{n}(P_i - \mu)(P_i - \mu)^T \cdots\cdots(2\text{-}49)$$

3. 将第二帧点云按位姿矩阵T转换为Q′，并计算在栅格化 P 的概率密度函数：

$$p(Q_i') = \frac{1}{(2\pi)^{D/2}\sqrt{|\Sigma|}}\exp\left(-\frac{(Q_i'-u)^T\Sigma^{-1}(Q_i'-u)}{2}\right) \cdots\cdots(2\text{-}50)$$

$$\Psi = \prod_{i=1}^{n} p(Q_i') \cdots\cdots(2\text{-}51)$$

4. 则求解位姿就是求解最大化Ψ对应的位姿：

$$\max_T \Psi = -\min_T \log(\Psi) = \min_T \sum_{i=1}^{n}\frac{(Q_i'-u)^T\Sigma^{-1}(Q_i'-u)}{2} \cdots\cdots(2\text{-}52)$$

5. 这是一个典型的非线性最小二乘问题，可以使用高斯牛顿法和 L-M 法进行求解。

LOAM（Lidar Odometry and Mapping）：在该方法中，为点云中每个点计算曲率，并根据曲率大小从点云中提取出两类特征点：边点（Edge Point）和面点（Planar Point）。当两帧点云之间的位姿变换正确时，边点应该在另一帧点云的某条边上，而面点应该在另一帧点云的某个面上。但是，由于位姿估计不准确，边点到边和面点到平面的仍存在一定距离，由此可以构造残差项和：

$$\sum_i d_{e,i} + \sum_i d_{p,i} \cdots\cdots(2\text{-}53)$$

上式中d_e和d_p分别是边点到边的距离和面点到平面的距离，其对应的计算公式如下：

$$d_e = \frac{|(Q_i'-P_j)\times(Q_i'-P_k)|}{|P_j-P_k|} \cdots\cdots(2\text{-}54)$$

其中，Q_i'是点云Q中的边点经过待优化变量T得到的三维点，而P_j、P_k是点云P中和Q_i'相同线数且距离最近的两个点。而d_e表示即为Q_i'到直线P_jP_k的距离。

$$d_p = \frac{|(Q_i'-P_j)((P_j-P_k)\times(P_k-P_m))|}{|(P_j-P_k)\times(P_k-P_m)|} \cdots\cdots(2\text{-}55)$$

其中，Q_i'是点云Q中的面点经过待优化变量T得到的三维点，而P_j、P_k、P_m是点云P中和Q_i'相同线数且距离最近的三个点。而d_p表示即为Q_i'到平面$P_jP_kP_m$的距离。之后，可以使用非线性优化的手段，不断迭代减小残差项和，最终求解出相对位姿T。

2.7.2 点云去畸变

在实际的数据采集场景中，由于机械式激光雷达不停旋转时，载体也在不断运动，造成在同一帧的点云并不是都在一个激光雷达坐标系中，从而出现了点云畸变的问题。

要想解决点云畸变的问题，要对每个点进行区别对待，即需要计算每个点相对于同一帧的第一个点的相对时间和相对位姿，通过位姿变换的方式，将其统一到同一帧初始时刻的激光雷达坐标系。

首先是相对时间的问题。如果点云数据每个点都带有时间戳，可以直接同一帧的第一个点相减，得到相对时间。如果没有携带时间戳信息，我们也可以通过计算相对旋转角度，再通过雷达转速计算出相对时间。例如，第一个点的坐标为P_0，点云中存在一点为P_{i}，可以计算相对水平角度为 $\arctan(P_{\mathrm{i,x}}/P_{\mathrm{i,y}}) - \arctan(P_{0,\mathrm{x}}/P_{0,\mathrm{y}})$，然后乘以转速$\frac{L_t}{2\pi}$，得到相对时间t。其中$L_t$表示旋转一圈花费的时间，对于 10HZ 的激光雷达来说，$L_t = 0.1s$。

对于相对位姿的问题，一般采用匀速模型或 IMU 辅助计算这两种方式。

匀速模型：如果没有 IMU 作为辅助的话，往往会利用载体的匀速模型进行去畸变。即，认为在同一帧的激光雷达旋转期间，载体是在做匀速运动。假定采集到的为第 i 帧点云，则其采集的时间在t_i和t_{i+1}之间。并已知了T_0、T_1、…、T_i，通过匀速模型可以计算：

$$\mathrm{T}_{i+1} = \mathrm{T}_i * (T_{i-1}^{-1} * \mathrm{T}_i) \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2\text{-}56)$$

对于时刻 t，在平移上进行线性插值，在旋转上进行球面线性插值：

$$\mathrm{Trans}_t = \mathrm{Trans}_i + \frac{t-\mathrm{t}_i}{\mathrm{t}_{i+1}-\mathrm{t}_i}(\mathrm{Trans}_{i+1} - \mathrm{Trans}_i) \quad \cdots\cdots\cdots\cdots (2\text{-}57)$$

$$\mathrm{Q}\{\mathrm{R}_t\} = \frac{sin((1-t)\theta)Q\{R_i\}+sin(t\theta)Q\{R_{i+1}\}}{sin\theta} \quad \cdots\cdots\cdots\cdots\cdots (2\text{-}58)$$

IMU 辅助计算：在有 IMU 作为额外传感器参与时，可以使用 IMU 的数据对载体位姿进行更加准确的估计。在已知T_i的前提下，可以从 IMU 的T_j开始，使用 IMU 中值积分公式，计算之后 IMU 的每一帧位姿$\mathrm{T}_j \sim \mathrm{T}_{j+n}$。在已知每个点相对时间 t 的基础上，可以找到其相邻的两帧 IMU 位姿，然后再上述公式进行位姿的

插值，平移向量的线性插值和四元数的球面线性插值。

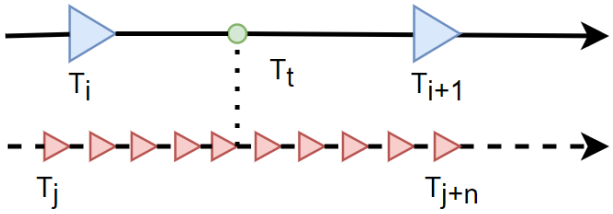

图 2.3 IMU 插值示意图

2.7.3 点云下采样算法

点云数据包含大量的点，并且具有无序性，在进行邻域搜索等点云操作时，需要付出很高的计算成本。为增加点云处理的效率，常见的解决方案就是点云下采样。通过点云下采样，在大量的点云中提取出适量的关键点上，将对于原始点云的操作转换为对关键点云的操作，降低整体的计算量。

常见的点云下采样方法有以下几种：

最远点采样法（Farthest Point Sampling，FPS）[48]：以点与点之间的欧式距离为参考依据，尽可能的覆盖点云空间的所有点，相比于以往的随机采样，可以更好控制点云的密度，更好的体现整体的点云分布情况。FPS 的采样过程可以描述为以下步骤：

假设需要从n个点组成的点云P中选取m个点作为采样后的点云 Q。

1. 随机挑选一个点P_i作为初始点，加入到点云$Q = \{P_i\}$。

2. 在剩余的n-1 个点的点云 P 中寻找一个和P_i最远的点P_j，并加入到点云$Q = \{P_i, P_j\}$。

3. 在剩余的n-2 个点的点云 P 中，依次计算到点云 Q 中点的距离，选择其中最短的距离作为该点到点云 Q 的距离，从而选择出点云 P 中最远的点P_k，并加入到点云$Q = \{P_i, P_j, P_k\}$。

4. 重复步骤（3），直到点云 Q 中点的数量达到 m。

体素化网格下采样（VoxelGrid Sampling）[49]：使用体素化网格的方法实现下采样，将点云划分到三维栅格中，每个栅格中只选取一个点的方式，从而减少整

体的点云数量。具体采样过程可以描述为以下步骤：

1. 设置栅格 X 轴、Y 轴和 Z 轴的大小。

2. 遍历整个点云，找到点云中 X 轴、Y 轴和 Z 轴的最小值分别为 $X_{min}, Y_{min}, Z_{min}$。

3. 遍历整个点云，根据栅格大小和各轴最小值计算所属栅格的编号，并加入到该栅格中。

4. 使用一个点代表栅格内的所有点，常见的表示方式为栅格的中点、栅格内点的重心或者随机挑选一个点。之后，将每个栅格中的代表点，组织成新的点云。

曲率下采样[50]：在进行点云下采样的时候，考虑点云局部的曲率信息。在曲率变化大的部分，进行较多的采样；而在曲率较为平缓的部分，进行较少的采样。曲率下采样的的过程可以描述为以下步骤：

1. 遍历整个点云，搜索每个点的 K 邻域，用以计算每个点的曲率信息。

2. 设置曲率阈值，将曲率大于阈值的点认为是特征明显的部分，而曲率小于阈值的点认为是特征较为不明显的部分。

3. 设置采样数，在特征明显的地方选取较多的点，在特征不明显的地方选取较少的点。

2.7.4 KD-Tree

K-Dimensional Tree（KD-Tree）[51]是一种在 K 维空间分割数据的数据结构，常常用于在多维数据中进行近邻查找。在本文中，在进行点云配准等算法时，用来加速对近邻点的查找速度。

构造 K-DTree，类似于构建二叉搜索树。在 K 维的数据中，选择一维 d 作为分割维度，将整体的 K 维数据按照 d 维从小到大排序，再进行一分为二的划分，分为两个子树，使得左子树中的 d 维值均小于右子树中 d 维值。在 K 维中选取一种 d 维作为划分维度的策略是，统计数据在每个维度的方差，选择最大方差所对应的维度，保证数据分割可以获得较好的平衡。而划分子树的策略是，使用 d 维的中值数据，将两个子树的大小保持均衡，防止树在构建时产生退化的问题。像这样对划分好的子树进行进一步的构建，直到该子树里只包含的一个数据。

在 KD-Tree 上，查找最近点有以下步骤：

步骤（1）：针对于查询数据 Q，从 KD-Tree 的根节点开始比较，一直向下进行访问，直到访问到叶子节点。在进行数据 Q 和节点的比较时，是值在该节点所分割时使用的维度上，比较数据 Q 和节点在该维度的大小关系，如果数据 Q 较小，则访问左子树，否则访问右子树。到达叶子节点后，计算数据 Q 和该节点的距离，并记录下来作为最邻近点和最小距离 Dist。

步骤（2）：在 KD-Tree 上进行回溯操作，判断在步骤（1）遍历中是否存在未访问的子树，仍有可能含有更近的近邻。如果数据 Q 与父节点的另一分支的距离小于最小距离 Dist，则进入到该子树内，进行步骤（1）的操作；否则，说明该分支不包含更近的近邻，继续执行回溯操作，直到回溯到根节点为止。

而 K 近邻查找，与最近邻查找类似，只是需要维护额外的大小为 K 的堆，方便进行元素比较、插入和删除操作。值得注意的是，邻域查找的最坏复杂度为 $O(n^{1-\frac{1}{k}})$，所以 KD-Tree 的高维查找效率并不是十分高效。

2.8 常见软件库

2.8.1 ROS

机器人操作系统（Robot Operating System，ROS）[52]是在机器人领域十分常见的开源操作系统，其在通用操作系统（Windows、Linux 等等）之上，为开发者提供底层驱动管理，进程间通信、功能包管理共用等功能。繁荣的 ROS 社区已经提供了十分稳定易用的功能包，如可视化功能包（RVIZ）等。

ROS 组织结构有：节点（Node）和节点管理器（ROS Master）。节点为可以独立运行的、可以完成一项或者多项功能的可执行文件。节点管理器全局只有一个，节点可以有多个。节点管理器为节点提供注册和查询功能，辅助不同节点间完成通信任务。

ROS 常用的通信机制有：主题（Topic）通信、服务（Service）通信。主题通信有两种操作，分别为发布（Publish）和订阅（Subscribe）。发布话题时，将特定的消息类型发布到特定的话题名上，会在节点管理器中完成注册。而发布话题是，从特定的话题名中获取发布的消息，在其回调函数中完成消息的后续操作。需要的注意的是，发布和订阅的话题名和消息类型一致时才能完成消息的传递。

这是一种消息的单向传递形式。而服务通信则是一种双向消息传递方式。节点在节点管理器中注册特定形式的服务。而在进行服务请求时，向其发送一个特定形式的请求（Request）消息，而在消息得到处理后，节点会返回的一个消息给发送方，从而完成一次消息的交互。这两种消息通信都是可以跨进程的，帮助开发者可以十分方便的编码出稳定的多进程程序。

ROS Bag 是 ROS 中提供的消息保存的格式，可以方便的完成消息的保存和回放。保存消息时，使用 rosbag record 命令对特定的主题进行消息保存，并存储于本地。而播放消息时，可以使用 rosbag play 可以方便的对特定的主题按时间顺序发送消息，从而将离线数据简单的提供给其他功能包。

2.8.2 PCL

点云库（Point Cloud Library，PCL）[53]是一个专注于处理点云数据的常用开源算法库。它是跨平台的，可以应用在多种操作系统上，提供点云过滤、点云配准、关键点提取、点云分割、采样一致性、点云读入和存储、KD-Tree 等功能，并且可以和 OpenMP、TBB、CUDA 等底层高性能开发库联合使用，利用 CPU 和 GPU 完成点云数据的并行的处理，进而加快数据的处理，为一些应用的实时性提供了底层保证。它可以高效地应用于点云可视化、激光遥感测绘、虚拟现实、人机交互和机器人等领域。

在 PCL 的内部实现中，大量使用 SIMD 优化，点云被封装到了智能指针中，可以完成资源的高效传递和及时释放。PCL 的点云类型是使用 C++模板（Template）实现，从而开发者可以自定义点云类型，利用 PCL 提供的函数，进而可以方便的完成自定义点云类型和 ROS PointCloud2 消息相互转换。

2.8.3 TensorRT

TensorRT[54]是一个用于在 NVIDIA GPUs 上加速高性能深度学习推理的 C++库，可以让开发人员方便高效地搭建低延迟和高吞吐率的深度学习部署。TensorRT 的高性能推理在边缘计算、自动驾驶或者数据中心等平台有着广泛的应用。TensorRT 生态十分健康，支持 TensorFlow、Pytorch 等目前主流的深度学习框架。

TensortRT 提供 C++ API 和 Python API，主要是在 GPU 上进行高性能推理加速。神经网络在训练阶段，包含前向传播和反向传播两个步骤，由前向传播算出损失函数的值，再经过反向传播调整神经网络中的权重。而神经网络在推理阶段，只包含前向传播阶段，使用的是训练时的模型权重。模型参数量巨大或者计算设备算力有限，就会导致推理过程速度变慢。所以，为了提高推理速度，研究人员提出了许多轻量神经网络，比如 MobileNet[55]、Shufflenet[56]等。而 TensorRT 将深度网络模型进行逐层解析和映射，将原本在深度学习框架中训练的模型转换到 TensorRT 中，通过优化策略进行部署加速。常见的优化手段有：层间融合和张量融合和数据精度校准。在原本的模型中，CUDA 核心启动和每一层的输入输出占据很长的时间，可以通过合并若干个常用的层为新的层，减少核心多次启动和输入输出的时间代价；在原本的模型中，为保证权重调整的精确，训练时常用的为 32 为浮点数精度，但是在推理过程中，可以适当降低权重精度，比如 FP16 或者 INT8，加速数据之间运算速度。

2.9 本章小结

本章介绍了本文所涉及的理论知识。首先，介绍了坐标系的整体定义和所使用的的符号及其含义说明，方便第三章的行文。之后，针对位姿变化中十分重要的旋转表示，分别介绍了旋转矩阵、欧拉角和四元数表示法，及其对应的常用公式及其作用。接着，从 SLAM 问题引入了最小二乘问题，介绍了其常用的非线性优化手段，高斯牛顿法、列文伯格-马夸尔特方法等等。之后，便是本章节的重点内容，位姿图的表示及其求解方法。首先使用位姿表示构造出整个位姿图的待优化问题，而位姿图包含一元边和二元边，可以分别进行一元边位姿修正和二元边的位姿修正，推导出了其观测量相对位姿的雅克比矩阵，结合马氏距离到最小二乘问题的转换，可以直接使用非线性优化的手段进行求解，为第三章的因子图求解提供了理论基础。再之后，是本文所使用到的传感器 LiDAR 和 IMU，分别介绍了其工作原理和对应的数学模型，也介绍了本文中使用的 LiDAR 和 IMU 的外参标定方法。针对激光雷达采集得到的点云数据，又展开介绍了其配准、去畸变、下采样算法和 KD-Tree。最后，针对本文第三章的具体代码实现，介绍了所使用的的软件库：ROS 机器人操作系统用以整个工程的构建和消息通信；PCL 点

云库用以工程中的点云操作；TensorRT 为 C++深度学习推理库，为项目中的语义分割网络提供底层支持。

第 3 章 多传感器融合的 SLAM 设计与实现

整体上，本文设计的 SLAM 系统采用图优化作为后端优化方式，相比于滤波器方法，可以考虑更多的状态关联，并且易于多种传感器之间的数据融合。传感器使用了 LiDAR、IMU 和 RTK。其中 LiDAR 相比于相机不易受光照影响，测量误差更小；而 IMU 可以高频率的输出载体的加速度、角速度信息，可以辅助 LiDAR 去畸变和配准，并且在激光雷达失效的场景下，也可以临时充当里程计，避免整个系统的崩溃。此外，本系统使用 RTK 的输出作为载体轨迹的真值，用作评价系统的定位质量。

3.1 因子图结构

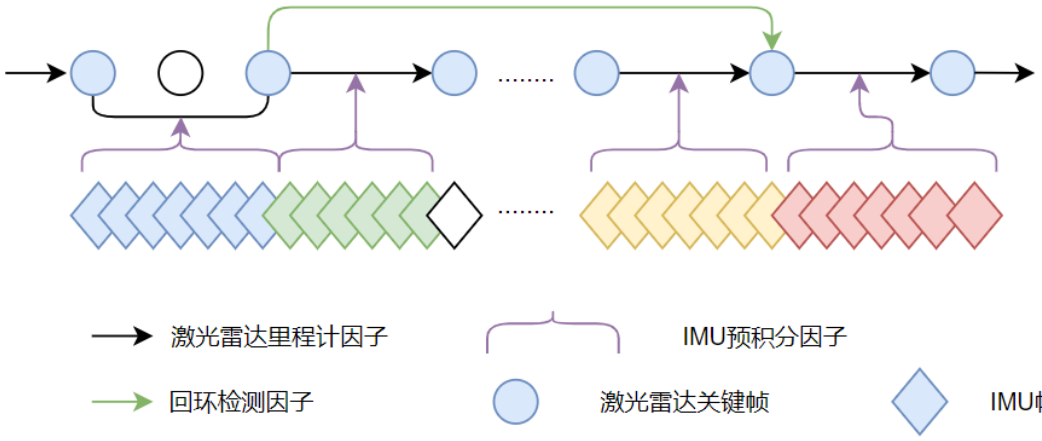

图 3.1 系统因子图示意图

如上图中所示，IMU 帧是反映载体不同时刻的加速度、角速度信息；而激光雷达帧反应载体不同时刻的待优化的位姿。使用 IMU 预积分和点云配准的方式可以分别对两帧间的相对位姿进行观测，并且可以提供各自的观测噪声，添加到因子图中。另外，回环检测也是 SLAM 中十分重要的的一环，可以帮助系统减少累积误差，也会加入到因子图中。

然而，如果我们把每一帧的结果都放入到因子图中的话，整个系统的待优化变量就会增加许多，进而每一次优化时的速度将会减慢，系统实时性将会无法保证。如果仅将少部分的帧放入到因子图中，整个系统的约束减少，噪声将会被放大，系统的准确性就会无法保证。于是，我们使用选取关键帧的方式来平衡实时

性和准确性。考虑到在实际的数据中时，车辆若保持低速情况下，可以降低帧的加入频率，因为前后帧的相对位姿不大，累计误差在接受范围以内；而车辆若在高速情况下，则需要增加帧的加入频率，否则，前后帧之间的距离过远，点云配准算法的效果将会大大下降。综上所述，本文采用了基于相邻帧位姿的关键帧选取手段，即对于已有关键帧x_i的情况下，新保存的帧为x_{i+1}，可以计算相对位姿T_i^{i+1}，若其小于一定阈值，则将x_{i+1}视为关键帧，相关的观测量会加入到因子图中，并且参与到后续的关键帧选取；若T_i^{i+1}大于一定阈值，则舍弃x_{i+1}帧，继续计算后续帧和x_i的相对位姿。在本文中，选取关键帧的阈值为相对位置小于 0.5m，相对旋转小于 5 度。

3.2 IMU 预积分因子

3.2.1 IMU 预积分

根据 IMU 不断输出的加速度和角速度信息，通过不断的积分操作，可以得到载体不同时刻的位姿。但是 IMU 本身的零偏存在随机游走的噪声，并且重力对 IMU 的旋转角 Yaw 是不可观的。如果仅依靠 IMU 进行导航，其定位误差将会随时间呈指数递增，无法在工程中直接使用。所以 IMU 通常会和其他传感器来避免累积误差。在本文中，IMU 仅用于估计较短时间内（100ms）的位姿变化，并将其加入到优化框架中。

在进行 IMU 位姿积分时，积分项中包含 IMU 坐标系绝对位姿的旋转矩阵。而在进行非线性优化时，该旋转矩阵是在不断变化的，积分项也在不断变换，所以需要不断的进行积分运算，难以保证实时性。于是，本文采用了 IMU 预积分来减少积分项的多次重复积分，其特点是将 IMU 坐标系的绝对位姿转换为了相对于上一帧 IMU 坐标系的相对位姿，再积分得到帧间的相对位姿。

由 IMU 的测量模型，使用欧拉积分，可以得到 IMU 的离散状态更新方程：

$$\begin{cases} p_{i+1}^{B_k} = \hat{p}_i^{B_k} + v_i^{B_k}\Delta t + \frac{1}{2}(R\{q_i^{B_k}\}(\hat{a}_{i+1} - b_a - \eta_a) - R\{q_i^{B_k}\}^T g)\Delta t^2 \\ v_{i+1}^{B_k} = v_i^{B_k} + (R\{q_i^{B_k}\}(\hat{a}_{i+1} - b_a - \eta_a) - R\{q_i^{B_k}\}^T g)\Delta t \\ q_{i+1}^{B_k} = q_i^{B_k} \otimes q\{(\hat{\omega}_{i+1} - b_g - \eta_g)\Delta t\} \end{cases} \cdot (3\text{-}1)$$

观察上述公式，可以发现其中仍然包含着变换的旋转矩阵$R\{q_i^{B_k}\}$，于是，可以去除和重力相关的项，定义预积分项：

$$\begin{cases}\alpha_{B_{k+1}}^{B_k} = p_{B_{k+1}}^{B_k} + \frac{1}{2}R\{q_{B_k}^w\}^T g\Delta t^2 \\ \beta_{B_{k+1}}^{B_k} = v_{B_{k+1}}^{B_k} + R\{q_{B_k}^w\}^T g\Delta t\end{cases} \cdots\cdots (3\text{-}2)$$

而加速度和角速度的零偏符合高斯分布，且协方差和时间成正比：

$$\begin{cases}b_{a,i+1} = b_{a,i} + \eta_{b_a}\Delta t \\ b_{g,i+1} = b_{g,i} + \eta_{b_g}\Delta t\end{cases} \cdots\cdots (3\text{-}3)$$

综上所述，可以得到 IMU 预积分项真值的离散更新方程：

$$\begin{cases}\alpha_{i+1}^{B_k} = \alpha_i^{B_k} + \beta_i^{B_k}\Delta t + \frac{1}{2}R\{q_i^{B_k}\}(\hat{a}_{i+1} - b_a - \eta_a)\Delta t^2 \\ \beta_{i+1}^{B_k} = \beta_i^{B_k} + R\{q_i^{B_k}\}(\hat{a}_{i+1} - b_a - \eta_a)\Delta t \\ q_{i+1}^{B_k} = q_i^{B_k} \otimes q\{(\hat{\omega}_{i+1} - b_g - \eta_g)\Delta t\} \\ b_{a,i+1} = b_{a,i} + \eta_{b_a}\Delta t \\ b_{g,i+1} = b_{g,i} + \eta_{b_g}\Delta t\end{cases} \cdots\cdots (3\text{-}4)$$

其中，初始化预积分项为：$\alpha = \beta = \mathbf{0}$。

然而，传感器本身的数据是存在噪声的，需要将积分项中标称项和误差项分离出来。标称项是测量值中不包含噪声的状态，而误差项是其中包含噪声项的状态。它们符合下述公式：

$$\begin{cases}\alpha^{B_k} = \hat{\alpha}^{B_k} + \delta\alpha^{B_k} \\ \beta^{B_k} = \hat{\beta}^{B_k} + \delta\beta^{B_k} \\ q^{B_k} = \hat{q}^{B_k} \otimes q\{\delta\theta^{B_k}\} \\ b_a = \hat{b}_a + \delta b_a \\ b_g = \hat{b}_g + \delta b_g\end{cases} \cdots\cdots (3\text{-}5)$$

在得到 IMU 预积分的定义后，可以推导出标称值离散更新方程、误差值离散更新方程、协方差更新方程和雅可比矩阵更新方程[47]。

（1） 标称值离散更新方程

根据 IMU 预积分项真值的离散更新方程，可以得到：

$$\begin{cases}\alpha_{i+1}^{B_k} = \alpha_i^{B_k} + \beta_i^{B_k}\Delta t + \frac{1}{2}R\{q_i^{B_k}\}(\hat{a}_{i+1} - b_a)\Delta t^2 \\ \beta_{i+1}^{B_k} = \beta_i^{B_k} + R\{q_i^{B_k}\}(\hat{a}_{i+1} - b_a)\Delta t \\ q_{i+1}^{B_k} = q_i^{B_k} \otimes q\{(\hat{\omega}_{i+1} - b_g)\Delta t\} \\ b_{a,i+1} = b_{a,i} \\ b_{g,i+1} = b_{g,i}\end{cases} \cdots\cdots (3\text{-}6)$$

（2）误差值离散更新方程

$$\begin{cases} \delta\alpha_{i+1}^{B_k} = \delta\alpha_i^{B_k} + \delta\beta_i^{B_k}\Delta t \\ \delta\beta_{i+1}^{B_k} = \delta\beta_i^{B_k} - R\{\hat{q}_i^{B_k}\}(\hat{a}_{i+1} - b_{a,i+1})^{\wedge}\delta\theta_i^{B_k}\Delta t - R\{\hat{q}_i^{B_k}\}\delta b_{a,i}\Delta t - R\{\hat{q}_i^{B_k}\}\eta_a\Delta t \\ \delta\theta_{i+1}^{B_k} = \delta\theta_i^{B_k} - (\hat{\omega}_{i+1} - b_g)^{\wedge}\delta\theta_i^{B_k}\Delta t - \delta b_{g,i}\Delta t + \eta_b\Delta t \\ \delta b_{a,i+1} = \delta b_{a,i} + \eta_{b_a}\Delta t \\ \delta b_{g,i+1} = \delta b_{g,i} + \eta_{b_g}\Delta t \end{cases} \cdots\cdots (3\text{-}7)$$

将上述方程组整理为：

$$\delta\mathbf{x}_{i+1} = (\mathrm{I} + \mathbf{F}_i\Delta t)\delta\mathbf{x}_{i+1} + \mathbf{G}_i\boldsymbol{\eta}\,\Delta t \cdots\cdots (3\text{-}8)$$

其中，

$$\mathbf{F}_i = \begin{bmatrix} 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & -\mathbf{R}\{\mathbf{q}_i^{B_k}\}(\mathbf{a}_{i+1} - \mathbf{b}_{ai+1})^{\wedge} & -\mathbf{R}\{\mathbf{q}_i^{B_k}\} & 0 \\ 0 & 0 & -(\omega_{i+1} - \mathbf{b}_{gi+1})^{\wedge} & 0 & -\mathbf{I} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{G}_i = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\mathbf{R}\{\hat{\mathbf{q}}_i^{B_k}\} & 0 & 0 & 0 \\ 0 & -\mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{bmatrix}$$

$$\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_a \\ \boldsymbol{\eta}_g \\ \boldsymbol{\eta}_{b_a} \\ \boldsymbol{\eta}_{b_s} \end{bmatrix}$$

（3）协方差矩阵更新方程：

根据上式和协方差定义得：

$$\Sigma_{i+1} = (\mathrm{I} + \mathbf{F}_i\Delta t)\Sigma_i(\mathrm{I} + \mathbf{F}_i\Delta t)^T + (\mathbf{G}_i\Delta t)Q(\mathbf{G}_i\Delta t)^T \cdots\cdots (3\text{-}9)$$

其中，Σ_i表示第 i 时刻的误差值δx协方差，Q是测量噪声的协方差。

（4）雅克比矩阵更新方程

根据上式子和导数定义得：

$$\mathrm{J}_{i+1} = (\mathrm{I} + \mathrm{F}_i\Delta t)\mathrm{J}_i \cdots\cdots (3\text{-}10)$$

其中，J_0初始化为单位矩阵。

3.2.2 IMU 残差项

在较短的 IMU 预积分的过程中，本文假设零偏是不变的，通过对 IMU 数据的不断处理，可以计算出帧间相对位姿。根据 IMU 预积分的定义及其相关方程，

可以推导出 IMU 预积分残差方程和雅可比方程[47]。

（1）IMU 预积分残差方程：

$$r_B = \begin{bmatrix} \delta\alpha^{B_k}_{B_{k+1}} \\ \delta\beta^{B_k}_{B_{k+1}} \\ \delta\theta^{B_k}_{B_{k+1}} \\ \delta b_a{}^{B_k}_{B_{k+1}} \\ \delta b_g{}^{B_k}_{B_{k+1}} \end{bmatrix} = \begin{bmatrix} R\{q^W_{B_k}\}^T\left(p^W_{B_{k+1}} - p^W_{B_k} - v^W_{B_k}\Delta t + \frac{1}{2}g\Delta t^2\right) - \hat{\alpha}^{B_k}_{B_{k+1}} \\ R\{q^W_{B_k}\}^T\left(v^W_{B_{k+1}} - v^W_{B_k} + g\Delta t\right) - \hat{\beta}^{B_k}_{B_{k+1}} \\ 2\{\hat{q}^{B_k}_{B_{k+1}}{}^* \otimes q^{W\,*}_{B_k} \otimes q^W_{B_{k+1}}\}_{xyz} \\ b_a^{B_{k+1}} - b_a^{B_k} \\ b_g^{B_{k+1}} - b_g^{B_k} \end{bmatrix} \tag{3-11}$$

其中，$r_B \sim \mathcal{N}(0, \Sigma_B)$，$\Sigma_B$是通过上述协方差更新方程计算的。

（2）IMU 残差项雅克比方程：

$$\mathbf{J}^{\mathbf{r}_B}_{\mathbf{x}_k\mathbf{x}_{k+1}} = -\partial x^{x_i x_i + 1}$$

$$\begin{bmatrix} \frac{\partial\delta\alpha^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{p}^W_{B_k}} & \frac{\partial\delta\alpha^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{v}^W_{B_k}} & \frac{\partial\delta\alpha^{B_k}_{B_{k+1}}}{\partial\delta\boldsymbol{\theta}^W_{B_k}} & \frac{\partial\delta\alpha^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{b}_{aB_k}} & \frac{\partial\delta\alpha^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{b}_{gB_k}} & \frac{\partial\delta\alpha^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{p}^W_{B_{k+1}}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{\partial\delta\boldsymbol{\beta}^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{v}^W_{B_k}} & \frac{\partial\delta\boldsymbol{\beta}^{B_k}_{B_{k+1}}}{\partial\delta\boldsymbol{\theta}^W_{B_k}} & \frac{\partial\delta\boldsymbol{\beta}^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{b}_{aB_k}} & \frac{\partial\delta\boldsymbol{\beta}^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{b}_{gB_k}} & \mathbf{0} & \frac{\partial\delta\boldsymbol{\beta}^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{v}^W_{B_{k+1}}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \frac{\partial\delta\boldsymbol{\theta}^{B_k}_{B_{k+1}}}{\partial\delta\boldsymbol{\theta}^W_{B_k}} & \mathbf{0} & \frac{\partial\delta\boldsymbol{\theta}^{B_k}_{B_{k+1}}}{\partial\delta\mathbf{b}_{gB_k}} & \mathbf{0} & \mathbf{0} & \frac{\partial\delta\boldsymbol{\theta}^{B_k}_{B_{k+1}}}{\partial\delta\boldsymbol{\theta}^W_{B_{k+1}}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{\partial\delta\mathbf{b}^{B_k}_{aB_{k+1}}}{\partial\delta\mathbf{b}_{aB_k}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{\partial\delta\mathbf{b}^{B_k}_{aB_{k+1}}}{\partial\delta\mathbf{b}_{aB_{k+1}}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{\partial\delta\mathbf{b}^{B_k}_{gB_{k+1}}}{\partial\delta\mathbf{b}_{gB_k}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{\partial\delta\mathbf{b}^{B_k}_{gB_{k+1}}}{\partial\delta\mathbf{b}_{gB_{k+1}}} \end{bmatrix} \tag{3-12}$$

3.3 语义 LiDAR 里程计因子

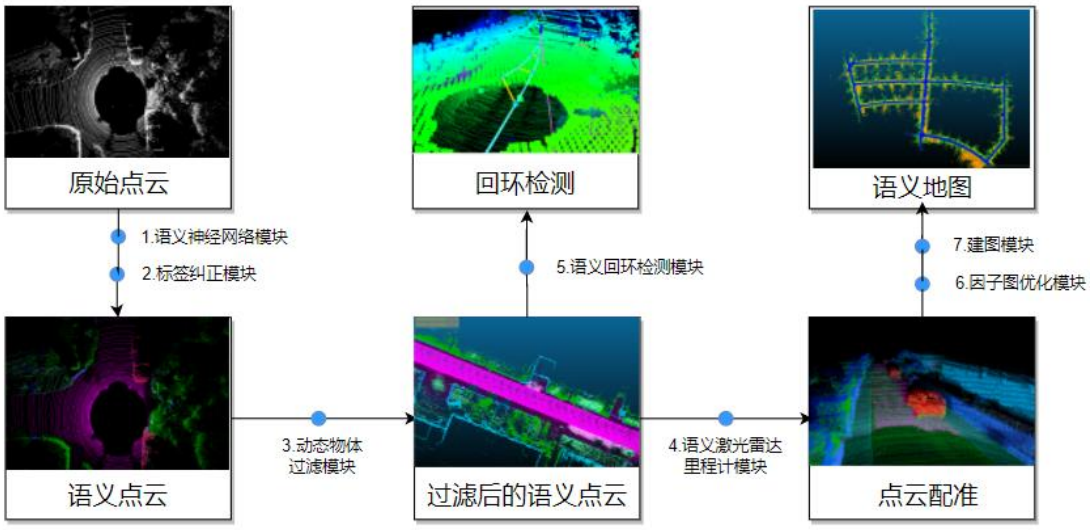

图 3.2 语义激光雷达里程计与回环检测流程图

本文提出的语义激光雷达里程计与回环检测银子的流程图，如上图所示。整个框架分为七个模块。首先是语义分割模块。原始的点云数据 P，经过 SPVNAS 神经网络的预测后，可以为点云中的每个点提供语义标签 L 及其标签置信度 Prob。之后，是标签纠正模块。经过神经网络预测的点云仍存在一定的标签错误率，使用基于聚类的标签纠正方法，得到更为准确的语义标签 L 及其标签置信度 Prob。然后，是动态物体过滤模块。通过对语义标签进行分类，完成点云中的动态物体滤除，得到静态场景的带有语义信息的点云数据，将其分别应用到语义 LiDAR 里程计模块和语义回环检测模块。语义 LiDAR 里程计主要负责点云配准工作，计算出相邻间的相对位姿，插入到因子图中作为两个节点之间的观测边。而语义回环检测模块主要负责点云的回环检测部分，经过 Semantic Scan Context 和语义 ICP 的处理后，将当前帧的点云和历史帧中的某一帧之间建立回环约束，用以消除两帧之间的累计误差，最终得到匹配帧间的相对位姿，插入到因子图中作为上述两个节点之间的观测边。之后，为因子图优化模块。使用 GTSAM 后端优化库求解融合多种约束的因子图，得到全局一致的位姿估计。最后，为语义模块。使用估计出来的位姿，逐帧将激光雷达坐标系下的点云转换到全局坐标系下，通过八叉树的语义处理，生成全局语义地图。

3.3.1 标签自纠正算法

虽然经过 SPVNAS 网络的预测，每个点获得一个类标签，但语义分割的结果仍然存在标签一定的错误率，需要处理语义标签错误的问题，因为它会影响后续点云配准过程的准确性。为了减少这些错误，本文使用基于欧几里德距离的聚类策略来校正类标签。基本原理是具有相同语义信息的对象往往出现在点云中的块区域中，我们可以根据周围点的语义分布重新计算网络预测的具有低置信度的点的标签。而距离置信度低的点越近，其标签的参考值就越大。换句话说，欧氏距离越小，点之间的标签信息就越一致。图 3.3 演示了标签校正的过程。

设 $P=\{P_1,P_2,\dots,P_n\}$为 LiDAR 采集的一帧点云，其中 P_i 为 P 中的一个点云点。经过 SPVNAS 处理的P，类别标签的可能性向量可以表示成下式：

$$L_i=\{p_{i,1},p_{i,2},\dots,p_{i,n}\} \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(3\text{-}13)$$

图 3.3 类别标签纠正示意图

其中，$p_{i,j}$表示第i个点为第j个类别的概率，n为 19，是类别的总数。$max(\cdot)$函数表示返回给定向量的最大值.如果$\max(L_i)$小于设置的阈值θ,使用公式（3-14）（3-15）根据周围点的分布权重，重新计算该点的概率向量，并且重新赋予该点的类别标签。

$$L_i' = \frac{1}{k+1}\left\{\sum_{P_j \in P_{local}} \frac{L_j}{\|P_i - P_j\|} + L_i\right\} \cdots\cdots (3\text{-}14)$$

$$p_{i,j}' = \frac{p_{i,j}}{\sum_{m=1}^{n} p_{i,m}} \cdots\cdots (3\text{-}15)$$

P_{local}是点 P_i 周围 k 个大于阈值θ的点云集合。其中，$p_{i,j}$, $p_{i,m} \in L_i'$, $p_{i,j}'$ 表示L_i'归一化后第 j 个类别对应的概率值。最后,L_i'中的$p_{i,j}$被重新赋予给第 i 个点。在具体实现时，本系统使用 KD-Tree 来保存点云，加速给定点的近邻查找速度。

3.3.2 动态物体过滤

数据采集过程中难免会在数据中记录一些动态对象，如行人、车辆等。这些不该出现在地图里的物体，不仅影响生成地图的精度，也会影响基于地图定位算法的准确性。大多数现有的 SLAM 系统并没有很好的考虑这个问题，他们都会事先假设环境是静态的。

在本文中，可以根据神经网络网络感知的结果过滤动态物体。具体来说，本文利用语义分割网络提供的标签信息来处理动态物体。定义 P_{piror} 表示当前帧点云，$P_{filtered}$ 表示标记为动态对象的点云，需要对其进行过滤。 P_{Map} 是过滤后用于建图的点云，可以通过以下公式计算：

$$P_{Map} = P_{prior} - P_{filtered} \cdots\cdots (3\text{-}16)$$

其中，$P_{filtered} = \{(x,y,z)|C_{x,y,z} \in D\}$，$C_{x,y,z}$ 表示该点的类别，D 表示需要过滤的动态物体的标签集合。图 3.4 描述了过滤动态物体的前后对比图。

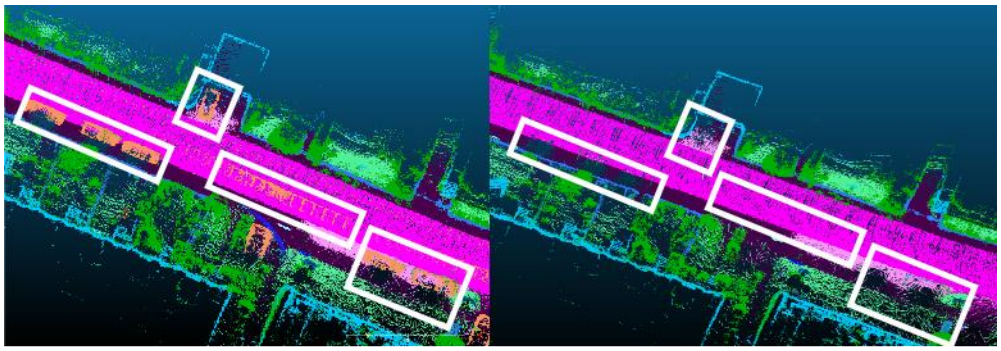

图 3.4 动态物体过滤可视化

3.3.3 特征提取

本文采用了 LOAM 中使用的特征提取方法，通过计算局部区域点的曲率提取边缘和平面特征。除了提取几何特征之外，我们也能够通过语义分割网络处理获得每个点的语义信息。在提取过程中，本文定义$S = \{P_i | \max(L_i) > \alpha\}$，表示类别标签的置信度大于$\alpha$的点云集合。$S$中点$P_i$对应的曲率$c_i$可以使用下述公式计算。

$$c_i = \frac{1}{n}\left\|\sum_{j=1}^{n}(P_i - P_j)\right\| \cdots\cdots(3\text{-}17)$$

其中，如果c_i小于阈值β，为面点；如果c_i大于阈值β，为边点。

本文定义$F_k = \langle F_k^p, F_k^e \rangle$为 k 时刻使用语义和曲率信息提取的特征点集合。其中，F_k^p表示提取的面点集合，F_k^e表示提取的边点集合。在提取特征点时，综合考虑了它们的几何特征和语义特征。换句话说，需要的匹配对应该来自两次连续扫描之间的同一个对象。使用这样的特征点来寻找对应关系，不仅提高了准确率，而且缩小了潜在的候选对象，提高了匹配效率。

3.3.4 语义损失函数

LOAM 或 LIO-SAM 中提到的特征提取过程依赖环境中的几何信息，并使用这些几何特征来建立点云之间的关联。本文使用语义信息扩展该方法以提高特征匹配的准确性和效率。详细说明如下：

设F_k^e和F_k^p分别为 k 帧生成的特征边点集合和特征面点集合，它们在 k+1 帧的对应关系建立为F_{k+1}^e和F_{k+1}^p。假设因子图中节点 N_{k+1}的初始位姿为 T_{k+1}，其

中T_{k+1}作为 4×4 的变换矩阵。通过$T_k T_{k+1}^{-1}$的相对位姿，我们可以将F_{k+1}^e和F_{k+1}^p重新投影到时间k，这样我们就可以得到$\hat{F}_{k+1}^e$和$\hat{F}_{k+1}^p$。点云帧之间的语义损失函数可以表示为：

$$l_e(P_i^e) = \varphi_e(P_i^e, P_u^e, P_v^e) \cdot \frac{|(P_i^e - P_u^e) \times (P_i^e - P_v^e)|}{|P_u^e - P_v^e|} \cdots\cdots\cdots\cdots\cdots\cdots (3\text{-}18)$$

$$l_p(P_i^p) = \varphi_p(P_i^p, P_u^p, P_v^p, P_w^p) \cdot \frac{\left|\begin{matrix}(P_i^p - P_u^p) \\ (P_u^p - P_v^p) \times (P_u^p - P_w^p)\end{matrix}\right|}{|(P_u^p - P_v^p) \times (P_u^p - P_w^p)|} \cdots\cdots\cdots\cdots (3\text{-}19)$$

其中，$P_i^e \in \hat{F}_{k+1}^e$，$P_u^e \in F_k^e$，$P_v^e \in F_k^e$，$P_i^p \in \hat{F}_{k+1}^p$，$P_u^p \in F_k^p$，$P_v^p \in F_k^p$，和 $P_w^p \in F_k^p$. $\varphi(*)$ 为权重函数。通常，周围点的标签离点P_i越近，权重越大。在这种情况下，还需考虑标签差异越大，权重越小。最终结果是减少了前端里程计不匹配对姿势优化的影响。 图 3.5 显示了提取的边缘和平面特征点以及构建的帧之间的约束关系。

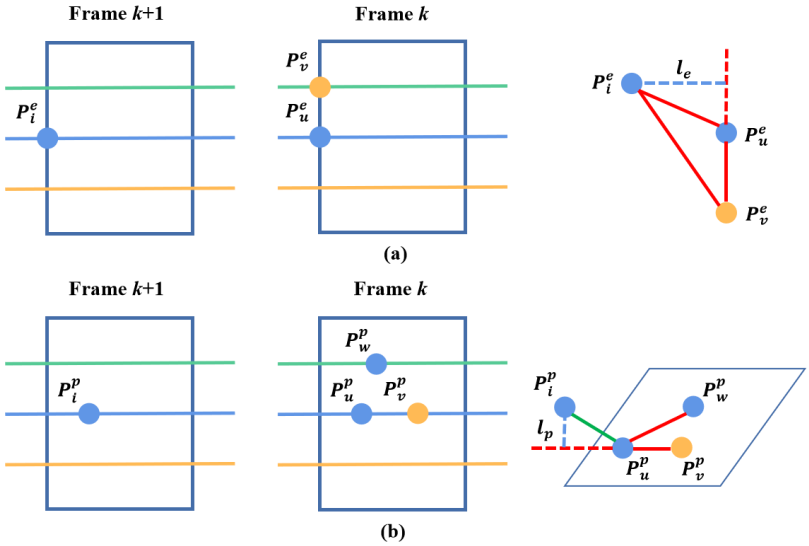

图 3.5 帧间特征点和约束关系的比较：（a）边缘特征点损失函数计算，（b）平面特征点损失函数计算。不同的颜色点代表不同的语义标签。

设我们数据集上 SPVNAS 网络的 19×19 混淆矩阵为 C。其中，$C_{i,j}$ 表示预测标签为 i，而真实标签为 j 的概率。

$$\varphi_e(P_i^e, P_u^e, P_v^e) = \alpha \min\left(C_{i,m}, C_{u,m}, C_{v,m}\right) \cdots\cdots\cdots\cdots\cdots\cdots (3\text{-}20)$$

$$\varphi_p\left(P_i^p, P_u^p, P_v^p, P_w^p\right) = \alpha \min\left(C_{i,n}, C_{u,n}, C_{v,n}, C_{w,n}\right) \cdots\cdots\cdots\cdots (3\text{-}21)$$

其中，$m = argmax(L_i, L_u, L_v)$，$n = argmax(L_i, L_u, L_v, L_w)$，$argmax(*)$是一个

函数，它根据每个点的预测值计算最可能的标签。假设所有点的标签为 n 或 m，计算每个点对应概率的最小值作为加权项。为防止整体误差过小而影响优化质量，阈值设为α。

在边点和面点选取上一帧的对应点时，如果按照 LOAM 的选取办法，当前帧的点和上一帧的点间距过大，容易造成匹配出现错误，所以我们选取出将上一帧中最近的五个点，算出其三个特征向量，然后在其中构造出若干点，再参与到后续的计算中。但是，这仍然是一种 Scan to Scan 的方法，前后两帧的位姿变化较大时或两帧之间的匹配对数减少，也会大大影响优化结果。于是，本文进而采用的为 Scan to Map 的方法，即让当前帧和上一帧以及之前若干帧形成的子图进行配准，增加配准的鲁棒性。

于是整体的点云配准的主要流程为：

（1）局部地图构建。以上一帧为基点，选取历史帧中距离最近的帧和时间最近的帧，再对构建帧进行稀疏化，剔除相邻较近的帧，将每一帧的边点和面点加入到一起，分别构建为边点局部地图和面点局部地图。

（2）当前帧投影。将 IMU 预积分的结果作为当前帧到上一帧的初始相对位姿，将当前帧提取的边点和面点投影到上一帧坐标系下。

（3）对当前帧中一个给定的特征点，查找出局部地图中最近的 5 个相近类别的点，计算这个五个点的协方差矩阵，使用奇异值分解的方式，求得三个特征值及其对应的特征向量。若该特征点为边点，进入步骤（4）；若为面点，则进入步骤（5）。

（4）对于边点，判断该特征向量是否为“一大两小”。若是，则存在匹配边，且较大的特征向量对应的为该边的法向量，边的中心两侧分别取点，得两个点作为匹配点，使用上述公式；若否，则不存在匹配关系。

（5）对于面点，判断该特征向量是否为“一小两大”。若是，则存在匹配面，且较小的特征向量对应的为该面的法向量，在面的中心附近分别去三个点，且保证不共线的情况下，作为匹配点，使用上述公式进行运算；若否，则不存在匹配关系。

对于，匹配的特征点集合，使用 Levenberg-Marquardt (L-M) 方法通过最小化以下方程来求解最优变换：

$$\min_{T_k T_{k+1}^{-1}} \{\sum_{p_i^e \in \hat{F}_{k+1}^e} l_e\left(P_i^e\right) + \sum_{p_i^p \in \hat{F}_{k+1}^p} l_p\left(P_i^p\right)\} \cdots\cdots\cdots\cdots\cdots\cdots(3\text{-}22)$$

使用 LM 求解该非线性最小二乘问题的关键在于损失函数相对位姿的雅克比矩阵。下面将分别推导出边点损失函数和面点损失函数对应的雅克比矩阵。

最后，可以得到相对位姿，$\Delta T = T_k T_{k+1}^{-1}$，并将其分配给因子图中位姿节点 k 和 k+1 之间的观察边。

3.4 语义回环检测因子

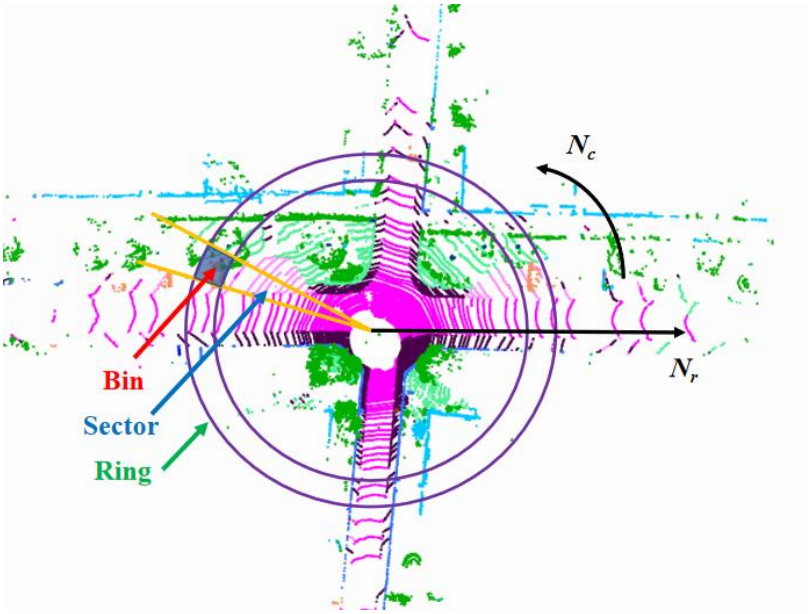

图 3.6 语义辅助 Scan Context 编码过程

回环检测是 SLAM 中一个非常重要且具有挑战性的问题。在系统的关键帧选取后，会有单独的线程，在历史帧队列中搜索是否存在回环帧。其中，回环帧，顾名思义，即和当前帧所处位置大致相同的帧。通过建立当前帧和回环帧的联系，可以减少这两帧间的累计误差。没有回环检测的 SLAM，相当于是一直在新的线路。而带有回环检测的 SLAM，在走回原先轨迹时，会意识到场景的重复性。

回环检测算法的主要分为两个步骤：第一步，通过匹配算法确定当前帧和历史帧中存在回环关系的候选子集。第二步，在候选子集中，选取最合适的帧构成回环检测。

3.4.1 语义点云描述子

基于欧氏距离的回环检测方法，会在累计误差较大时失效，无法确定匹配关

系；而基于局部或全局几何特征描述子的回环检测方法，如 Scan Context 等，很容易受动态场景的影响，导致回环检测的错误匹配。考虑到人类在生活中常常会利用语义信息来完成场景识别，判断自己是否走回了原地。在这种启发下，本文在 Scan Context 中添加了语义信息并提出了一个新的全局描述子（Semantic Scan Context，SSC）。

首先，将点云帧编码成为 SSC。在构建 SSC 时，将三维点云以扇区块的形式组织。在点云俯视图中，径向按照距离均分，称为环（Ring）；并且按照角度划分，称为扇形（Section）。由此，点云在俯视图中被分割成了一个个扇形区域，如图 5 所示，这种小区域称为 bin。令 N_c 和 N_r 是语义 Scan Context 图像的列数和行数，d_{max} 是点云的最远距离。而在本实验中，N_c 设置为 60，N_r 设置为 20，d_{max} 设置为 80。那么语义辅助扫描上下文图像 Q 可以描述如下：

$$Q = \cup_{i\in N_r, j\in N_c} w(s_{i,j}) \cdot \max z(P_{i,j}) \cdots\cdots\cdots\cdots\cdots\cdots(3\text{-}23)$$

其中,$w(s_{i,j})$ 是块(i,j)对应的语义信息权重。$P_{i,j}$为 block(i,j)对应的坐标信息。同时$\mathrm{z}(\cdot)$是用来获取点$P_{i,j}$高度信息的函数。

在给定两帧语义辅助扫描上下文图像Q_i和Q_j的情况下，本文使用余弦向量来估计。具体为，计算Q_i和Q_j中相同列之间的余弦值，一减去之后的值，每列都相加到一起：

$$\mathrm{d}(Q_i, Q_j) = \frac{1}{N_r}\sum_{k=1}^{N_r}\left(1 - \frac{Q_i^k \cdot Q_j^k}{\|Q_i^k\|\|Q_j^k\|}\right) \cdots\cdots\cdots\cdots\cdots\cdots(3\text{-}24)$$

其中，Q_i^k表示Q_i中第 k 列所形成的向量。

但是，形成回环检测的两帧点云之间往往存在一定偏航角。针对这种情况，可以不断调整其中一帧的初始匹配列。由此，可以两帧之间的相似度可以估计为：

$$\mathrm{d}(Q_i, Q_j) = \min_{n\in[N_r]}\left\{\frac{1}{N_r}\sum_{k=1}^{N_r}\left(1 - \frac{Q_i^k \cdot Q_j^{(k+n)\%N_r}}{\|Q_i^k\|\|Q_j^{(k+n)\%N_r}\|}\right)\right\}\cdots\cdots\cdots\cdots(3\text{-}25)$$

其中，$Q_j^{(k+n)\%N_r}$表示Q_j中以第 n 列为起始列的情况下第 k 列所形成的向量。

但是，这种两帧之间相似度计算复杂度为$\mathrm{O}(\mathrm{n}^2)$，时间代价比较高昂。因此，在实际的实验中，本文使用的是两步搜索的策略：首先，从语义辅助的扫描上下

文图像中提取 ring keys，因为这种 ring keys 不受偏航角的影响，可以将二维的 SSC 粗略的表示为一维向量，用作回环检测的粗匹配阶段。将每一帧的 ring keys 构建到一颗 KD 树中，可以快速方便的找到与当前帧近似的若干帧。在回环检测的细匹配阶段，使用上式计算这些候选语义辅助扫描上下文图像的相似度得分，并将得分最高的图像对应的帧检测为回环帧。

在实际的数据测试中发现，由于 Semantic Scan Context 是基于单帧点云的描述子，便会出现因场景相似而误认为回环的情况。因此，在回环帧确定后，增加了使用语义 ICP 校验的环节。

3.4.2 语义 ICP

设通过 Semantic Scan Context 确定的回环检测匹配对为点云帧P和点云帧Q，使用 Semantic ICP 算法计算两帧之间相对位姿的步骤如下：

（1）确定初值。如果是合理的回环检测匹配对，两帧的距离应该比较接近，因此不妨设相对位姿的初始值为单位矩阵。

（2）语义 ICP 损失函数。点云帧Q经过待优化相对位姿T投影到点云帧 P 附近后，可以构建带有标签点与点的损失函数：

$$E = \sum_i w_i \left\| \mathrm{P}_i - T\mathrm{Q}_i \right\|_2 \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(3\text{-}26)$$

$$w_i^k = \rho_{Huber}((\mathrm{P}_i - T^k\mathrm{Q}_i)^2)H(\mathrm{P}_i, \mathrm{Q}_i) \cdots\cdots\cdots\cdots\cdots\cdots\cdots(3\text{-}27)$$

其中，P_i是Q_i最近邻的点，可以使用 KD-Tree 快速查找出来。ρ_{Huber}为鲁邦核函数[57]：

$$\rho_{Huber}(\gamma) = \begin{cases} 1, & if\ |\gamma| < \delta \\ \delta|\gamma|^{-1}, & otherwise \end{cases} \cdots\cdots\cdots\cdots\cdots\cdots\cdots(3\text{-}28)$$

$H(\mathrm{P}_i, \mathrm{Q}_i)$表示两点标签的匹配度：

$$H(\mathrm{P}_i, \mathrm{Q}_i) = \begin{cases} max(Prob_{P_i}, Prob_{Q_i}),\ if\ Label_{P_i} = Label_{Q_i} \\ min(Prob_{P_i}, Prob_{Q_i}),\ if\ Label_{P_i} != Label_{Q_i} \end{cases} \cdots\cdots(3\text{-}29)$$

在标签相同时，使用较大的概率，成为重点优化的项；而在标签不同时，表明可能存在无匹配的情况，使用较小的概率。

（3）最小二乘问题求解。使用 LM 算法对上述损失函数进行优化，得到相对位姿T。若 T 小于一定的阈值，表明该回环检测匹配对是合理的；否则，则不认为是回环。在本文中，该阈值为两帧间相对位移 5m。

最后，如果在第 i 帧和第 j 帧之间存在回环对，则可以将观察边添加到因子图中。

3.5 语义地图的构建

在 SLAM 系统的运行中，载体的位姿估计和地图构建都是持续生成的。随着运行时间的推移，所生成的关键帧点云队列和全局地图将会越来越到大，对于计算机的内存容量提出了较大的需求。

所以，为了减少系统的内存消耗，做出了以下的改进：

（1）关键帧的点云队列并不需要时刻都保存在内存中，因为激光雷达里程计使用的为当前帧和历史帧构建的局部地图。所以，采用滑动窗口的方式，将滑动窗口内的关键帧点云保存在内存中，而滑动窗口之外的关键帧点云保存在硬盘中。

（2）为减少全局地图保存所使用的内存消耗，本系统将全局地图最终保存为存储效率高的带有语义信息的八叉树地图（OctoMap）。

语义八叉树地图的构建过程可以如下描述：

（1）得到第一个关键帧特征点云时，使用体素化栅格进行下采样，保存为初始的全局地图；

（2）得到后续的关键特征点云时，先使用体素化栅格进行下采样，计算该帧点云在全局坐标系下的位置，并进行点云的旋转平移，之后进行栅格点云的合并，保证一个栅格内只保留栅格的中心点，而栅格的语义标签则使用投票法计算栅格内最多语义标签类型。

（3）重复步骤（2），直到系统运行结束。最后将栅格地图使用八叉树算法进行融合和语义表示，并保存至硬盘中。

3.6 本章小结

本章节是本文的核心部分，主要介绍了整个 SLAM 系统的工作流程。本章节从第二章的位姿图部分作为切入点，引出了系统所使用的的因子图，分别包含 IMU 预积分因子、语义激光雷达里程计因子和回环检测因子。首先是 IMU 预积分因子，介绍了其出现的动机，从 IMU 量测模型开始进行推导，得到一系列的

离散更新方程，推导出 IMU 预积分的定义，结合 IMU 的测量值推导出 IMU 的残差项，并求解出 IMU 残差项相对于位姿的雅克比矩阵。之后是语义激光雷达里程计因子部分，给出了结合语义信息的里程计流程图，对其中的标签纠正、动态物体过滤、特征提取和语义损失函数等模块进行了详细的介绍。紧接着介绍了语义回环检测因子的相关内容，由常用回环检测方法的不足引出语义点云描述子和语义 ICP 结合的检测方法，先使用语义点云描述子匹配算法 提出候选回环帧，使用语义 ICP 进行后续确认。在本章节的最后，对语义地图的构建进行了说明，考虑到边缘设备内存有限的情况，做了针对性的优化。

第 4 章 实验与分析

在本章节中，我们使用实验室设备采集的校园数据，另外结合常用的 KITTI 数据集，设置了多组相关对比实验，从而验证了本文算法的可行性。

4.1 实验设置

在本小节中，将介绍实验中使用的实验传感器和平台。图 4.1 显示了用来采集 JLU 校园数据集的传感器。它由 Velodyne 的 HDL-32E LiDAR 传感器、3DM-GX5 惯性测量单元 (IMU) 传感器和 Trimble BD982 GNSS 接收器模块组成。 LiDAR 和 IMU 记录的数据用于生成全局地图。同时，我们也记录了 Trimble BD982 GNSS 生成的 RTK 位姿作为车辆的位姿真值，用以评估系统位姿估计的准确性。同时，RTK 可以为激光雷达和 IMU 提供统一的授时，保证数据采集的时间戳的一致性。

(a) (b) (c)

图 4.1 实验中使用的传感器设置：(a) Velodyne HDL-32E LiDAR，(b) 3DM-GX5 IMU，和 (c) 带有天线的 Trimble BD982 GNSS 接收器模块。

图 4.2 带有传感器的采集平台（大众途观）

以上所有传感器都集成到大众途观移动平台中，如图 4.2 所示。激光雷达放置在车顶的自制铝制框架上，以获得更好的视野。惯性测量单元 (IMU) 位于 LiDAR 正下方 17 厘米处。 GNSS 移动台放置在途观的后备箱中。另外，在车顶行李架周围放置三个 GNSS 天线，沿横向和纵向形成两条直线，以获取车辆的

位姿信息。

本实验分为数据采集和算法验证两部分。在数据采集时，使用的为搭载 Intel i7 CPU 的工控机。使用网线连接 Velodyne 的激光雷达，通过读取对应端口的 PCAP 数据，转换为 PCD 格式数据，最终保存为 sensor_msgs/PointCloud2 消息格式；使用 USB 连接 IMU，通过解析 CAN 数据报文，得到载体的角速度和加速度，并保存为 sensor_msg/IMU 消息格式；而使用网口连接 RTK 设备，通过解析 NMEA 数据报文，得到载体的经纬度，并保存为 sensor_msgs/ NavSatFix 消息格式。以上三种消息格式最终会写入到 ROSBAG 文件中，保存至硬盘中。

而进行算法验证时，使用的为一台服务器，其搭载两颗 Intel Xeon E5-2680 处理器，NVIDIA GTX1070 显卡，8*16G DDR3 内存条，运行 Ubuntu 18.04 操作系统。

4.2 数据集介绍

本系统在 KITTI 数据集和吉林大学校园数据上进行了实验，定性和定量地比较出本文提出的方法与 LeGO-LOAM 和 LIO-SAM 的建图效果。

KITTI Dataset: KITTI 是经典的自动驾驶数据集，广泛用于视觉里程计和 3D 对象检测任务。本实验使用来自 KITTI 原始 11 个序列 (00-10) 来构建全局地图，其中包含建图所需的点云、IMU 数据和位姿真值。每个序列的扫描帧数、轨迹长度和回环情况如表 1 所示。

表 4.1 KITTI 数据集细节

序列	帧数	轨迹长度 (米)*	存在回环 (是/否)
00	4541	-	是
01	1101	2453	否
02	4661	5067	是
03	801	-	否
04	271	393	否
05	2761	2205	是
06	1101	1232	是
07	1101	694	是
08	5171	3222	是
09	1591	1705	是
10	1201	919	否

吉林大学校园数据集: 我们还利用自动驾驶平台为途观大众开发了一个更

具挑战性的数据集，其中包含吉林大学的动态场景。 JLU Campus Dataset 包含五个数据序列，提供点云、IMU 数据用于建图和 RTK 数据用于精度评估。 JLU Campus 数据集的详细信息如表 2 所示。

表 4.2 吉林大学校园数据细节

序列	帧数	轨迹长度 (米)	存在回环 (是/否)
JLU_042800	3676	766	是
JLU_042801	1646	382	否
JLU_042802	7591	1829	是
JLU_050500	7763	1970	是
JLU_050501	10605	3221	是

4.3 实验结果与分析

4.3.1 评价指标

本文在轨迹对比实验中，采用的评价指标是 KITTI 数据集中的 RPE（Relative Pose Error），即相对位姿误差。假设算法估计位姿序列：P_0，P_1，…,P_n，且$P_i \in SE(3)$，而真值位姿序列：Q_0，Q_1，…,Q_n，且$Q_i \in SE(3)$。同时假设估计位姿和真值位姿数量相同且时间戳对齐。

$$E_{rot} = \frac{1}{n}\sum_{(i,j)\in[0,n]} \frac{Rot((P_j \ominus P_i) \ominus (Q_j \ominus Q_i))}{Length(i,j)} \cdots\cdots (4\text{-}1)$$

$$E_{trans} = \frac{1}{n}\sum_{(i,j)\in[0,n]} \frac{Trans((P_j \ominus P_i) \ominus (Q_j \ominus Q_i))}{Length(i,j)} \cdots\cdots (4\text{-}2)$$

其中，$Length(i,j)$表示固定的距离,在本文中为$[100, 200, \ldots, 800]$，$\ominus$表示两帧之间相对位姿，$Rot(\cdot)$, $Trans(\cdot)$分表示提取$SE(3)$中的旋转量和平移量。所以下述表格表示的旋转误差和平移误差，而实际的含义为每米产生的旋转误差（deg/m）和平均每百米的平移偏差（%）。

本文在回环检测实验中，采用的评价指标是 Precision-Recall（P-R）曲线。根据真值和预测值的情况，可以使用混淆矩阵进行表述。

表 4.3 真实值和预测值的混淆矩阵

	真实值为真	真实值为假
预测为真	TP	TN
预测为假	FP	FN

又有以下公式：

$$precision = \frac{\mathrm{TP}}{\mathrm{TP+FP}} \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \text{(4-3)}$$

$$recall = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \text{(4-4)}$$

通过调整阈值，得到不同的预测情况，进而计算出多组 $precision$ 和$recall$ 数值，通过比较不同曲线下的面积，便可以比较出分类方法的有效性。

4.3.2 实验一：KITTI 数据集

表 4.4 KITTI 数据集相对位姿误差

Sequence	LeGO-LOAM	LIO-SAM	LIO-CSI
00	-/-	-/-	-/-
01	0.0170/24.1277	0.0118/6.4019	**0.0114/6.0156**
02	0.0326/8.0823	0.0148/2.9097	**0.0100/2.2866**
03	-/-	-/-	-/-
04	0.0116/1.6022	**0.0108/1.4198**	0.0112/2.1670
05	0.0139/2.0792	0.0128/1.6909	**0.0072/1.2058**
06	0.0172/3.2835	0.0107/2.1312	**0.0090/1.5534**
07	0.0206/2.0103	**0.0162**/2.8697	0.0193/**1.4258**
08	0.0175/3.9161	0.0175/3.8697	**0.0172/3.2835**
09	**0.0148**/4.7493	0.0188/7.9129	0.0181/**3.7669**
10	0.0190/3.2337	0.0217/4.9693	**0.0160/2.5726**
Average	0.0182/5.8985	0.0150/3.7972	**0.0133/2.6975**

本实验将所提出的 LIO-CSI 方法与两种基于紧耦合 LiDAR 惯性里程计 LeGO-LOAM 和 LIO-SAM 进行了比较。首先在 KITTI 数据集上测试了这些方法，并使用平均相对姿态误差 (mRPE) 来评估结果。需要注意的是，表 4.4 中的

序列 00 和 03 没有结果。原因是 KITTI 官网中的序列 00 的 IMU 数据的时间戳无法匹配其 LiDAR 数据；序列 03 没有提供可供使用的 IMU 数据。

由表 3 显示，与 LeGO-LOAM 和 LIO-SAM 相比，本文提出的 LIO-CSI 方法在 KITTI 数据集上有很大的改进。三种方法中，LIO-CSI 的平均 RPE 结果是最好的。本文提出的方法的平均相对平移误差为 2.6975%。与 LeGO-LOAM 和 LIO-SAM 相比，本文的 LIO-CSI 方法分别将结果提高了 3.201% 和 1.0997%。本文的方法在大多数序列中都有效，尤其是在长距离序列中，例如 01、02、05 和 08。在超过 2 公里的序列上，与 LeGO-LOAM 和 LIO-SAM 相比，我们的平均相对平移误差分别提高了 6.3534%和 0.5202%。而以度/米为单位的平均相对旋转误差为 0.0133，与 LeGO-LOAM 和 LIO-SAM 相比，本文的方法分别减少了 26.9%和 11.3%的误差。

此外，一些细节也表明我们提出的方法在旋转估计方面的性能优于其他比较方法。图 4.3 和图 4.4 显示了 KITTI 数据集的序列 05 和 06 上的轨迹。与其他方法相比，本文的方法估计的轨迹数据最接近真实情况。序列 09 全长 1.7 公里，仅包含一个起点与终点重合的回环。在所有方法中，只有本文的方法成功检测到回环并生成了全局一致的地图，如图 4.5 所示。此外，图 4.7 中的彩色点云是本文的方法在序列 09 上生成的地图，蓝色的是由 LIO-SAM 生成的。本文的方法不仅在水平方向上找到起点和终点之间的一致性，而且在垂直方向上找到全局一致性。以上所有实验结果都说明了本文提出的方法的有效性，语义信息和 SLAM 框架的结合可以显著提高位置和旋转估计的准确性。

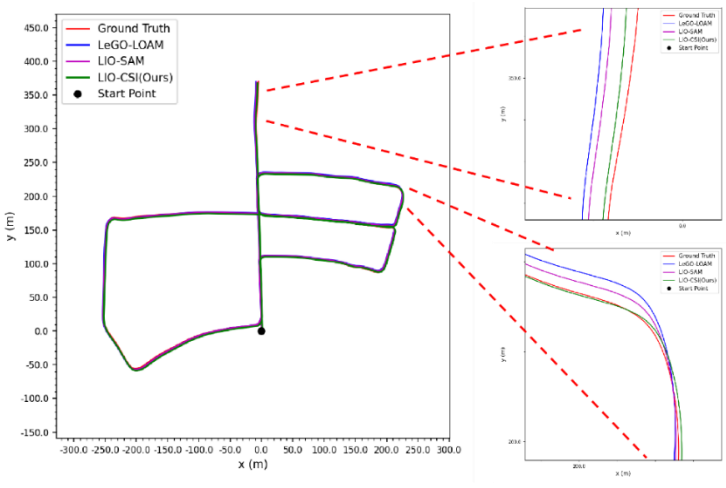

图 4.3 KITTI 数据集 05 序列 轨迹对比图

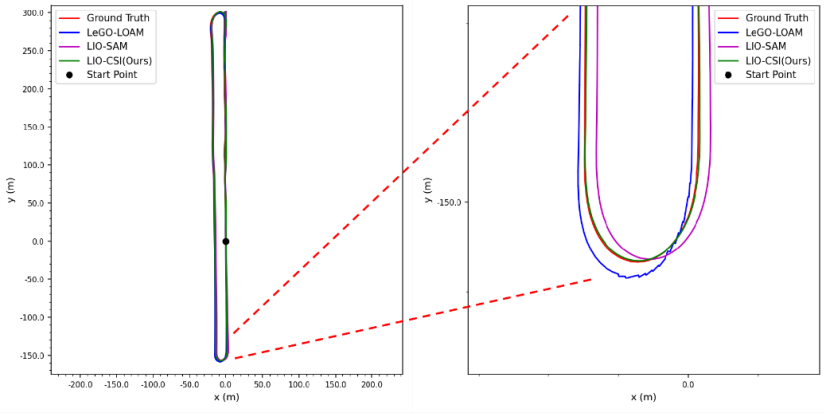

图 4.4 KITTI 数据集 06 序列 轨迹对比图

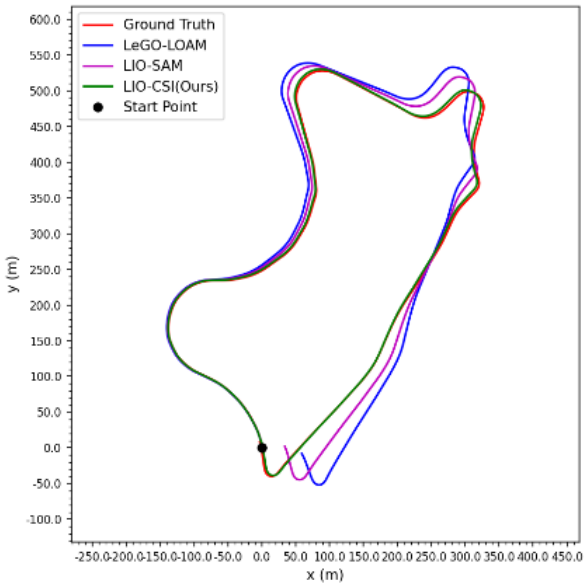

图 4.5 KITTI 数据集 09 序列 轨迹对比

图 4.6 KITTI 数据集 09 序列 语义地图

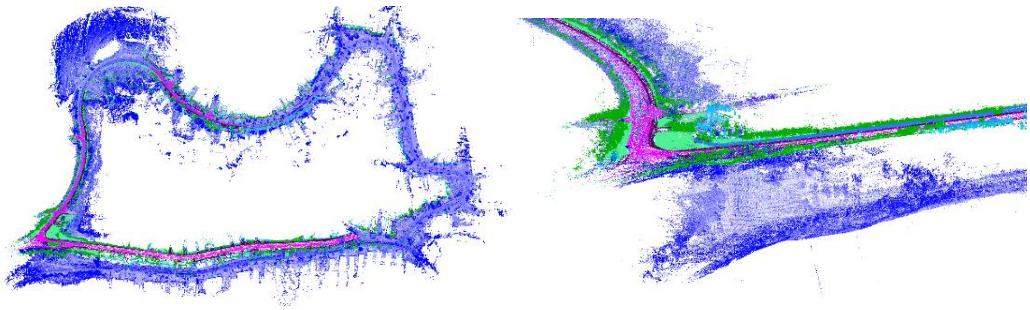

图 4.7 KITTI 数据集 09 序列 建图效果纵向对比图

4.3.3 实验二：消融实验

为了进一步验证将语义信息引入里程计和回环检测的好处，我们使用 KITTI 数据集进行了两组消融实验。本部分创建了一组实验来比较原始 LIO-SAM 与基于语义辅助里程计 (LIO-SAM-ODOM)的 LIO-SAM，另一组实验是将 LIO-SAM-ODOM 与本文提出的方法进行比较。它们之间的区别在于回环检测的策略。前者使用基于欧氏距离的回环检测方法，后者使用 3.4 节描述的语回环检测方法。

表 4.5 LIO-SAM 和 LIO-SAM-ODOM 相对位姿误差

Sequence	LIO-SAM	LIO-SAM-ODOM
01	0.0118/6.4019	**0.0114/6.0266**
02	0.0148/2.9097	**0.0129/2.7694**
04	0.0108/**1.4198**	**0.0074**/1.9278
05	0.0128/1.6909	**0.0069/1.0806**
06	**0.0107/2.1312**	0.0115/2.3138
07	**0.0162**/2.8697	0.0197/**1.4742**
08	**0.0175/3.8697**	0.0231/4.5125
09	0.0188/7.9129	**0.0167/4.8907**
10	**0.0217/4.9693**	0.0222/5.1645
Average	0.0150/3.7972	**0.0146/3.3511**

表 4.5 显示了 LIO-SAM 和 LIO-SAM-ODOM 的 mRPE 结果。本次测试旨在表明语义辅助里程计可以提高扫描匹配精度并提高 SLAM 框架的性能。在表中，本实验可以发现与 LIO-SAM 相比，LIO-SAM-ODOM 在平均 mRPE 结果方

面有显着改善。本实验还可以发现，LIO-SAM-ODOM 在距离超过 2 公里的序列上似乎更准确（参考表 4.5 中序列 01、02 和 05 的结果）。一种可能的解释是，相似几何特征的数量随着距离的增加而增加，但可以通过辅助语义信息建立约束来过滤一些相似的几何特征，减少错误匹配的发生。因此，可以得出结论，所提出的语义辅助里程计方法可以大大提高里程计的准确性。在远距离建图任务中效果更明显。

表 4.6 LIO-SAM-ODOM 和 LIO-CSI 相对位姿误差

Sequence	LIO-SAM-ODOM	LIO-CSI
02	0.0129/2.7694	**0.0100/2.2866**
05	**0.0069/1.0806**	0.0072/1.2058
06	0.0115/2.3138	**0.0090/1.5534**
07	0.0197/1.4742	**0.0193/1.4258**
08	0.0231/4.5125	**0.0172/3.2835**
09	**0.0167**/4.8907	0.0181/**3.7669**
Average	0.0146/3.3511	**0.0135/2.2537**

除了上述实验之外，我们还通过与 LIO-SAM-ODOM 的对比验证了本文提出的语义回环检测的有效性。实验是在具有回环的序列 02、05、06、07、08 和 09 上进行的。表 4.6 清楚地表明，使用语义辅助回环检测可以消除里程计的累积误差。除了序列 05 和 09，本文的方法得到的结果优于 LIO-SAM-ODOM。序列 05 的结果比 LIO-SAM-ODOM 略差。一个可能的原因是序列 05 的轨迹很复杂，因为它包含了很多回环。与欧式距离相比，语义辅助方法具有更高的计算复杂度，无法及时处理数据。而序列 09 的相对旋转结果略高于对比方法，但相对平移误差大大降低。

4.3.4 实验三：吉林大学校园数据集

为了模拟具有挑战性的动态场景，我们在拥有大量行人和车辆的 JLU 校园中收集了五个数据序列。数据采集过程中，车速保持在 30 公里/小时左右。

在本次实验中，结果的评估方式与在 KITTI 数据集上的评估方式相同。表 4.7 显示本文提出的 LIO-CSI 方法优于比较方法。042800、050500、050501 序

表 4.7 吉林大学校园数据相对位姿误差

Sequence	LeGO-LOAM	LIO-SAM	LIO-CSI
JLU_042800	0.0166/1.9177	0.0152/1.8044	**0.0142/1.6945**
JLU_042801	0.0111/**1.6337**	**0.0088**/3.3934	0.0095/3.6723
JLU_042802	0.0265/**3.2008**	**0.0260**/4.9488	0.0261/3.4284
JLU_050500	**0.0064/1.3017**	0.0115/4.1364	**0.0064/1.3017**
JLU_050501	0.0277/10.2588	0.0273/8.9888	**0.0272/5.1347**
Average	0.0177/3.6625	0.0178/4.6544	**0.0167/3.0463**

列的平均相对旋转误差和平移误差以及平均结果小于 LeGO-LOAM 和 LIO-SAM。与 LeGO-LOAM 和 LIO-SAM 方法相比，平均旋转误差分别降低了 5.6%和 6.2%，平均平移误差分别降低了 0.6162%和 1.6081%。

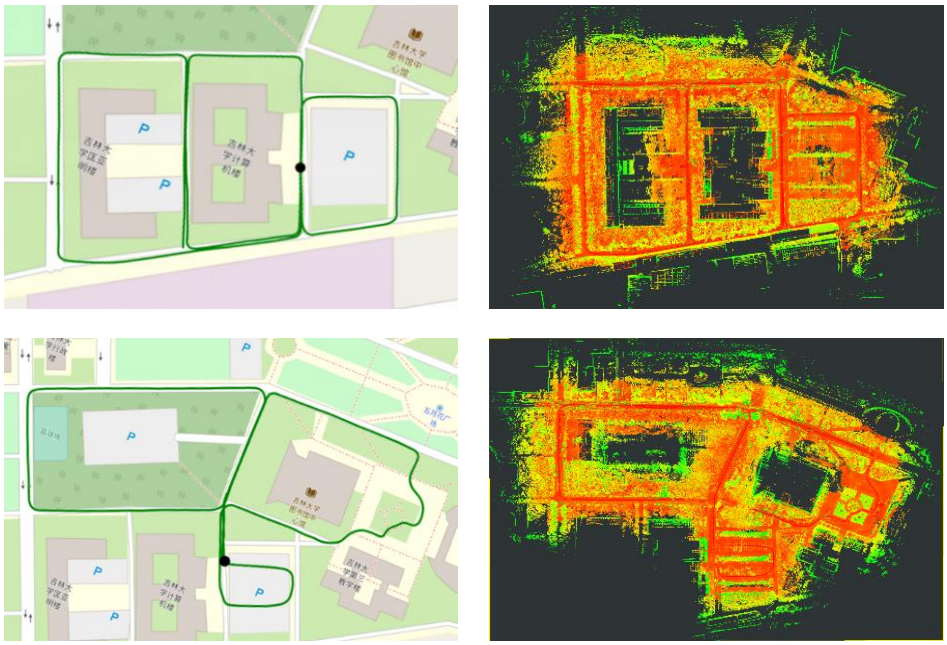

图 4.8. LIO-CSI 的 USGS National Map 图像和建图效果。左图中的绿色轨迹是系统估计的车辆位姿。

图 4.8 显示了本文的方法在 JLU Campus 数据集的实际效果。从生成的地图和可用的美国地质调查局国家地图图像的比较中，我们可以看出它们是高度一致的。值得注意的是，在表 4.6 中，LeGO-LOAM 的部分结果优于 LIO-SAM。这可能是因为 LeGO-LOAM 中的聚类策略在配准过程中提供了额外的标签信息，尤其是在动态场景中，特征的整合有助于获得更好的结果。该实验说明本文的方

法不仅提高了里程计的准确性，减少了累积误差，而且具有良好的鲁棒性和良好的泛化能力。

4.3.5 实验四：回环检测

本文通过计算两两关键帧间的欧拉距离来构建真值矩阵。假设L_i和L_j分别为同一序列的两个关键帧，如果$\|L_i - L_j\|_2 < 4m$，则认为这两帧构成回环；否则，为不构成。如图 4.9 所示，左侧为 KITTI 05 数据的轨迹真值，右侧为构建的真值矩阵。其中，可以看出，左侧的每一处回环序列，都在右侧体现为一条白线。表明，本实验构建的真值矩阵是可以代表场景中真实回环状态的。

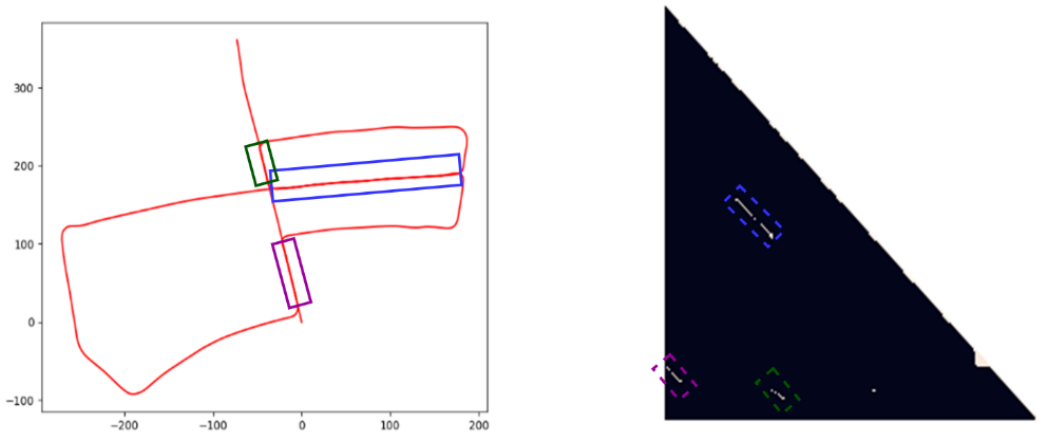

图 4.9 真实估计和真实矩阵对比图

通过两种不同的回环检测算法，Scan Context 和本文回环检测算法，可以分别计算出两帧之间的相似程度，使用[0,1]进行表示，并表示为矩阵的形式，如图 4.10 所示。图中第一列为构建的真值矩阵；第二列为 Scan Context 算法生成的相似度矩阵，除了真实的回环对应的白色外，还存在其他许多较浅的颜色，在阈值选择不恰当的情况下，很容易产生误判的情况；第三列为本文回环检测算法生成的结果，可以看到利用语义信息和语义 ICP 的处理，可以大大减少较浅的颜色，说明本文算法对回环的检测具有较高的区分度。图 4.10 中，每一行的结果分别对应 KITTI 数据集中的序列 00、序列 05 和序列 08。

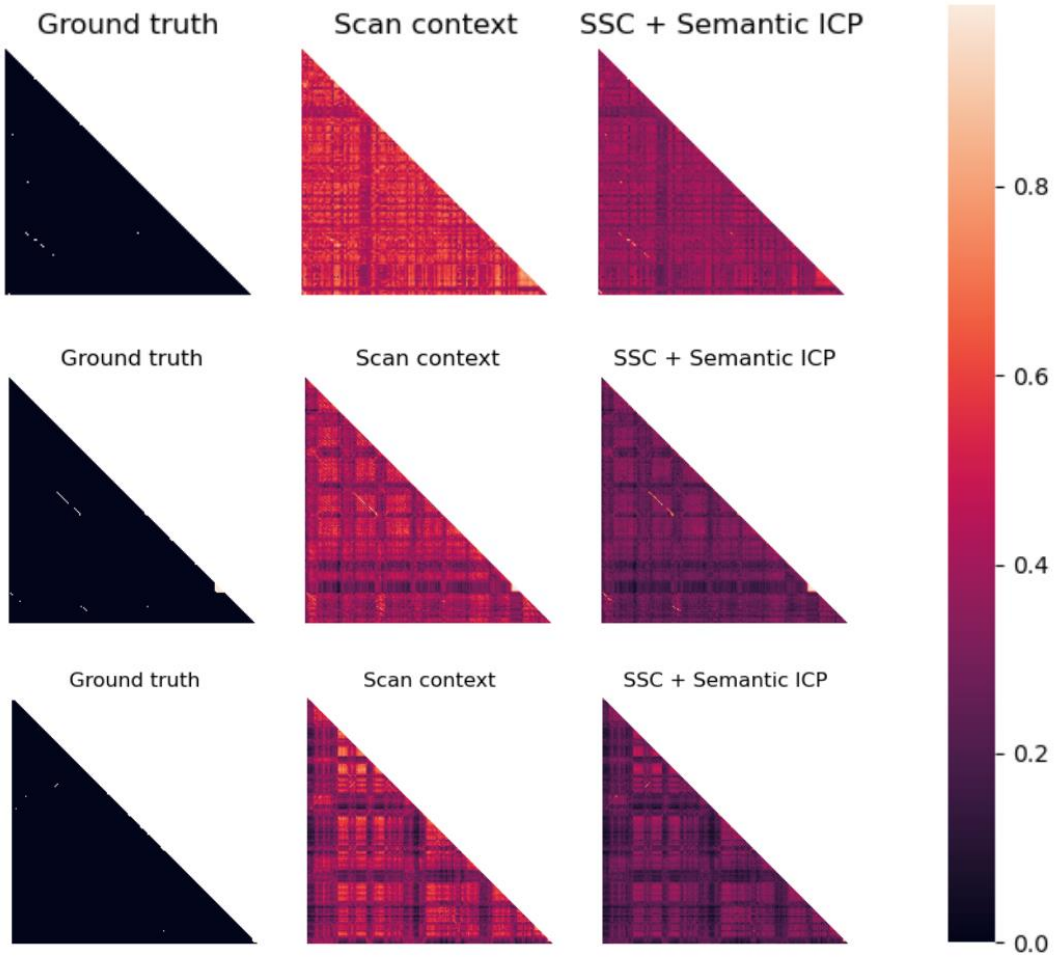

图 4.10 KITTI 相似度矩阵对比图

通过不断调整两个算法中的阈值，可以方便计算回环预测量，绘制成的 P-R 曲线。如图 4.12 所示，分别为 KITT 00、05 和 08 绘制的曲线。通过对比两条曲线，可以看出，在不同的阈值选取中，本文算法的准确率和召回率相比 ScanContext 算法均有较大提升。

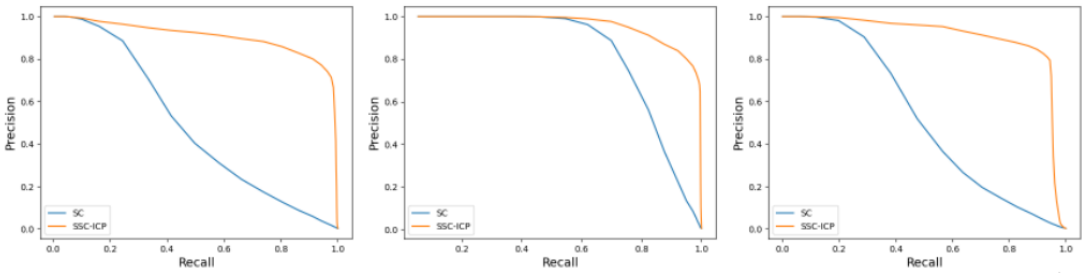

图 4.11 KITT P-R 曲线对比图

4.4 本章小结

本章节是本文的实验部分，通过设置各种对照实验，使用国内国外的数据集来说明本文算法研究的有效性。首先介绍了吉林大学校园数据集采集时使用的传感器型号和硬件平台，通过解析不同传感器的报文，打包成对应的消息格式，存储为 ROSBAG 文件。紧接着，介绍了实验将要用到的数据集：KITTI 数据集和吉林大学校园数据集。之后，介绍了实验使用的评价指标。设置实验一，在 KITTI 数据集上对比了 LeGO-LOAM、LIO-SAM 和本文算法的结果，表明本文算法整体具有一定的优越性；设置实验二，在 KITTI 数据集上分别将 LIO-SAM、LIO-SA-ODOM 和本文算法，两两对比，表明了出现在实验一中的结果，语义激光雷达里程计和语义回环检测均有贡献；设置实验三，在吉林大学校园数据集上，将 LeGO-LOAM 和本文算法做了对照实验，表明本文方法具有一定的鲁棒性，在国内校园场景仍有较好的表现；设置实验四，在 KITTI 数据上，将 ScanContext 算法和本文回环检测算法进行了定量定性的对比，在相似矩阵和 P-R 曲线都表现出本文算法有着较好的检测效果。

第 5 章 总结与展望

5.1 全文总结

定位建图作为自动驾驶中十分重要的一环，其定位精度和建图效果也深刻影响着其他重要环节。本文针对包含大量动态物体的复杂场景，设计并实现了融合激光雷达和 IMU 的 SLAM 方案。在激光雷达里程计中加入语义信息，去除动态物体并增加点云配准的精度。在回环检测方面，提出了语义点云描述子，结合描述子匹配算法和语义 ICP，提高回环检测的精度。使用实验室设备，设计并实现对校园数据的采集。结合校园数据和 KITTI 数据，本文设置多组对照实验，提出的 SLAM 算法相较于 LIO-SAM 和 LeGO-LOAM 具有较好的表现，并且具有一定的鲁棒性。

5.2 未来展望

近年来，激光雷达 SLAM 算法层出不穷，不断有优秀的算法推陈出新。本文仅仅尝试将语义信息和 SLAM 做出整合和优化，但是仍有许多方面值得继续探索。

首先，本文 SLAM 系统易受语义分割网络的推理速度和准确度的影响。使用相较而言，更加鲁棒的神经网络可以有效提高场景变换的适应性。未来，可以尝试通过模型压缩、蒸馏、剪枝等方法得到更加轻量的网络，加速神经网络的推理速度。

其次，语义激光雷达里程计部分只用到了语义信息和曲率信息。可以考虑抽象层级更高的，基于物体（Object-Based）的 SLAM 系统，尝试建立物体间的约束关系，针对物体整体做位姿优化。

最后，在回环检测方面，本文只是做了初步的尝试，可以完成在相差时间不大的相似场景的识别。然而，在现实中，完成回环可能已经过去很长时间，甚至以季度为单位。所以，在超长时间尺度下的回环检测问题是接下来可以考虑的研究目标之一。

参考文献

[1] 章军辉, 陈大鹏, 李庆. 自动驾驶技术研究现状及发展趋势 [J]. 科学技术与工程, 2020, 20(09): 3394-403.

[2] Durrant-Whyte H, Bailey T. Simultaneous localization and mapping: part I [J]. IEEE robotics & automation magazine, 2006, 13(2): 99-110.

[3] Bailey T, Durrant-Whyte H. Simultaneous localization and mapping (SLAM): Part II [J]. IEEE robotics & automation magazine, 2006, 13(3): 108-17.

[4] 王金强, 黄航, 郅朋, et al. 自动驾驶发展与关键技术综述 [J]. 电子技术应用, 2019, 45(6): 28-36.

[5] Mur-Artal R, Montiel J M M, Tardos J D. ORB-SLAM: a versatile and accurate monocular SLAM system [J]. IEEE transactions on robotics, 2015, 31(5): 1147-63.

[6] Mur-Artal R, Tardós J D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras [J]. IEEE transactions on robotics, 2017, 33(5): 1255-62.

[7] Qin T, Li P, Shen S. Vins-mono: A robust and versatile monocular visual-inertial state estimator [J]. IEEE Transactions on Robotics, 2018, 34(4): 1004-20.

[8] Zhang J, Singh S. LOAM: Lidar Odometry and Mapping in Real-time; proceedings of the Robotics: Science and Systems, F, 2014 [C]. Berkeley, CA.

[9] Chen X, Milioto A, Palazzolo E, et al. Suma++: Efficient lidar-based semantic slam; proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), F, 2019 [C]. IEEE.

[10] Xu W, Zhang F. Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter [J]. IEEE Robotics and Automation Letters, 2021, 6(2): 3317-24.

[11] Xu W, Cai Y, He D, et al. Fast-lio2: Fast direct lidar-inertial odometry [J]. arXiv preprint arXiv:210706829, 2021.

[12] Shan T, Englot B, Meyers D, et al. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping; proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), F, 2020 [C]. IEEE.

[13] Shan T, Englot B. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain; proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), F, 2018 [C]. IEEE.

[14] Shan T, Englot B, Ratti C, et al. Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping; proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), F, 2021 [C]. IEEE.

[15] Grisetti G, Stachniss C, Burgard W. Improved techniques for grid mapping with rao-blackwellized particle filters [J]. IEEE transactions on Robotics, 2007,

23(1): 34-46.
[16] Olson E B. Real-time correlative scan matching; proceedings of the 2009 IEEE International Conference on Robotics and Automation, F, 2009 [C]. IEEE.
[17] Konolige K, Grisetti G, Kümmerle R, et al. Efficient sparse pose adjustment for 2D mapping; proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, F, 2010 [C]. IEEE.
[18] Hess W, Kohler D, Rapp H, et al. Real-time loop closure in 2D LIDAR SLAM; proceedings of the 2016 IEEE international conference on robotics and automation (ICRA), F, 2016 [C]. IEEE.
[19] Besl P J, Mckay N D. Method for registration of 3-D shapes; proceedings of the Sensor fusion IV: control paradigms and data structures, F, 1992 [C]. Spie.
[20] Mendes E, Koch P, Lacroix S. ICP-based pose-graph SLAM; proceedings of the 2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), F, 2016 [C]. IEEE.
[21] Biber P, Straßer W. The normal distributions transform: A new approach to laser scan matching; proceedings of the Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat No 03CH37453), F, 2003 [C]. IEEE.
[22] Koide K, Miura J, Menegatti E. A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement [J]. International Journal of Advanced Robotic Systems, 2019, 16(2): 1729881419841532.
[23] 高翔. 视觉 SLAM 十四讲: 从理论到实践 [M]. 电子工业出版社, 2017.
[24] Kim G, Kim A. Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map; proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), F, 2018 [C]. IEEE.
[25] Zhang Y, Jin R, Zhou Z-H. Understanding bag-of-words model: a statistical framework [J]. International Journal of Machine Learning and Cybernetics, 2010, 1(1): 43-52.
[26] Steder B, Ruhnke M, Grzonka S, et al. Place recognition in 3D scans using a combination of bag of words and point feature based relative pose estimation; proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, F, 2011 [C]. IEEE.
[27] Rusu R B, Bradski G, Thibaux R, et al. Fast 3d recognition and pose using the viewpoint feature histogram; proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, F, 2010 [C]. IEEE.
[28] Wohlkinger W, Vincze M. Ensemble of shape functions for 3d object classification; proceedings of the 2011 IEEE international conference on robotics and biomimetics, F, 2011 [C]. IEEE.
[29] Noble W S. What is a support vector machine? [J]. Nature biotechnology, 2006, 24(12): 1565-7.
[30] Margineantu D D, Dietterich T G. Pruning adaptive boosting; proceedings of the ICML, F, 1997 [C]. Citeseer.

[31] Qi C R, Su H, Mo K, et al. Pointnet: Deep learning on point sets for 3d classification and segmentation; proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, F, 2017 [C].
[32] Qi C R, Yi L, Su H, et al. Pointnet++: Deep hierarchical feature learning on point sets in a metric space [J]. Advances in neural information processing systems, 2017, 30.
[33] Milioto A, Vizzo I, Behley J, et al. Rangenet++: Fast and accurate lidar semantic segmentation; proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), F, 2019 [C]. IEEE.
[34] Lawin F J, Danelljan M, Tosteberg P, et al. Deep projective 3D semantic segmentation; proceedings of the International Conference on Computer Analysis of Images and Patterns, F, 2017 [C]. Springer.
[35] Tchapmi L, Choy C, Armeni I, et al. Segcloud: Semantic segmentation of 3d point clouds; proceedings of the 2017 international conference on 3D vision (3DV), F, 2017 [C]. IEEE.
[36] Salas-Moreno R F, Newcombe R A, Strasdat H, et al. Slam++: Simultaneous localisation and mapping at the level of objects; proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, F, 2013 [C].
[37] Bowman S L, Atanasov N, Daniilidis K, et al. Probabilistic data association for semantic slam; proceedings of the 2017 IEEE international conference on robotics and automation (ICRA), F, 2017 [C]. IEEE.
[38] Parkhiya P, Khawad R, Murthy J K, et al. Constructing category-specific models for monocular object-slam; proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), F, 2018 [C]. IEEE.
[39] Yu C, Liu Z, Liu X-J, et al. DS-SLAM: A semantic visual SLAM towards dynamic environments; proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), F, 2018 [C]. IEEE.
[40] Humphreys J E. Introduction to Lie algebras and representation theory [M]. Springer Science & Business Media, 2012.
[41] Gilmore R. Baker - Campbell - Hausdorff formulas [J]. Journal of Mathematical Physics, 1974, 15(12): 2090-2.
[42] Wang G, Wei Y, Qiao S, et al. Generalized inverses: theory and computations [M]. Springer, 2018.
[43] Wang Y. Gauss–newton method [J]. Wiley Interdisciplinary Reviews: Computational Statistics, 2012, 4(4): 415-20.
[44] Moré J J. The Levenberg-Marquardt algorithm: implementation and theory [M]. Numerical analysis. Springer. 1978: 105-16.
[45] Dellaert F. Factor graphs and GTSAM: A hands-on introduction [R]: Georgia Institute of Technology, 2012.
[46] De Maesschalck R, Jouan-Rimbaud D, Massart D L. The mahalanobis distance [J]. Chemometrics and intelligent laboratory systems, 2000, 50(1): 1-18.
[47] Forster C, Carlone L, Dellaert F, et al. IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation, F, 2015 [C]. Georgia

Institute of Technology.
[48] Moenning C, Dodgson N A. Fast marching farthest point sampling [R]: University of Cambridge, Computer Laboratory, 2003.
[49] Orts-Escolano S, Morell V, García-Rodríguez J, et al. Point cloud data filtering and downsampling using growing neural gas; proceedings of the The 2013 International Joint Conference on Neural Networks (IJCNN), F, 2013 [C]. IEEE.
[50] 张娴, 张志毅, 田素垒, et al. 基于曲率分析的三次 Bezier 曲线采样方法的研究 [D], 2013.
[51] Ramasubramanian V, Paliwal K K. Fast k-dimensional tree algorithms for nearest neighbor search with application to vector quantization encoding [J]. IEEE Transactions on Signal Processing, 1992, 40(3): 518-31.
[52] Quigley M, Conley K, Gerkey B, et al. ROS: an open-source Robot Operating System; proceedings of the ICRA workshop on open source software, F, 2009 [C]. Kobe, Japan.
[53] Rusu R B, Cousins S. 3d is here: Point cloud library (pcl); proceedings of the 2011 IEEE international conference on robotics and automation, F, 2011 [C]. IEEE.
[54] Vanholder H. Efficient inference with tensorrt; proceedings of the GPU Technology Conference, F, 2016 [C].
[55] Howard A G, Zhu M, Chen B, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications [J]. arXiv preprint arXiv:170404861, 2017.
[56] Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices; proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, F, 2018 [C].
[57] Wang Y-G, Lin X, Zhu M, et al. Robust estimation using the Huber function with a data-dependent tuning constant [J]. Journal of Computational and Graphical Statistics, 2007, 16(2): 468-81.

致谢

时光荏苒，驹光过隙，我的研究生生涯即将画上句号。回忆往昔，不禁回想起，复试后联系导师的小心翼翼，殊不知这一刻起我的人生将要发生沧桑巨变。在吉林大学的三年时光，是我这一生都难以忘怀的回忆。一流的大学，一流的教育，让我的学习和生活都收获满满。

首先，我要由衷的感谢我的导师王刚老师。王老师一丝不苟、精益求精的精神极大的影响了我做人做事的方式方法，让我受益终身。在王老师的高效组织下，我有幸先后参与了多个实验室的重大项目和课题，提高了自己的眼界，锻炼了自己的代码实现能力，磨砺了和同学们相互合作的本领。王老师见闻广博、专业知识过硬，在我工作方向摇摆不定的关键时候，透彻地分析利弊，帮助我找到最后的方向。同时，在我论文选题、构思和撰写等方向给予了专业有效的指导。

其次，我要衷心的感谢实验室同门们，正是有了你们的有力支持，我才能够顺利的完成学业。感谢我的博士生学长们，徐谦学长、张同舟学长和侯明辉学长，在我论文的构思、撰写、格式纠正等方面提供了非常大的帮助。在进行论文实验的过程中，各位学弟学妹们也给予了我十分大的帮助。这三年来，十分庆幸可以和你们一路相伴，见证彼此的成长，共同为实验室发光发热。山高路远，我们顶峰相见。

最后，我要十分的感谢我的家人。你们一直无私付出，是我最为坚强的后盾。同时，在我择校择业时，也给了我极大的自由，让我可以尽可能地把未来活成自己想要的样子。十余年的漫漫求学路，十分幸运能够得到你们的的支持和付出。走入社会后，我也会努力工作，不忘你们的教诲与嘱托。

“唯有热爱，可抵岁月漫长”是我最喜欢的一句话，送给未来的自己。愿你千帆过尽，仍不负心中所爱。