High Speed Real-time Location Scheme Based on LEGO-LOAM

由 Georgia LI创建, 最后修改于七月 11, 2019

本文在调研现有适用于无人驾驶领域的SLAM方案的基础上，结合现有传感器条件及任务要求，从算法流程，代码架构，优化思路及可行性分析等方面给出了基于Lego-Loam算法的高车速实时定位方案。

一、SLAM导论

根据传感器的类型，现有SLAM方法一般分为Visual-SLAM（VSLAM）和LiDAR-SLAM（LSLAM）。尽管视觉传感器能带来大量的数据信息，在回环检测方面有较高的准确度，但视觉方法过分依赖于外界光强且受限于视野范围，使得其作为单一传感器进行SLAM时可靠性及稳定性都无法保证。相比而言，基于LiDAR的SLAM方法精度和准确度较高，且对障碍物阻挡及暗光环境适应性强，因而广泛应用于无人驾驶领域。

传统的LSLAM的传感器是2D-LIDAR，其解决方案的第一步是Scan-Matching，又称前端，即通过帧间数据的匹配实现机器人位置的初估计。传统的方法是运用ICP（Iterative Closest Point）持续迭代搜索整体误差最小的匹配结果，这种方法需要大量的数据以达到既定精度，因此计算成本也会随之增加。很多改进的ICP方法主要目标都是优化算法效率和精度。在此基础上，后端通过Scan-Matching给出的位姿实现Mapping，进而实现位置和地图的整体优化。2D-slam在前端可以取得不错的定位精度，但在进行回环检测时容易出现误匹配的情况，且2D信息无法感知垂直方向的信息，因而无法实现避障，感知等方面的需求。

3D-LiDAR可以有效获取3维激光数据，得到周围环境的点云信息，但由于其庞大的数量使得很多2D-SLAM解决方案无法实现较好的实时性，从而发展出一套基于特征点的匹配方法，它通过提取点云中的特征点来进行匹配从而有效降低了计算量。Figure 1为KITTI数据集上排名TOP5的前端里程计解决方案，除第4项外均是适用于3D-LiDAR，其中前两项均是基于LOAM算法的前端方案。LOAM方法选取选取边缘点和平面点作为特征点实现高频低精度的激光里程计，然后通过低频的建图进行修正以实现高精度的位姿估计（位置估计错误率均低于0.6%，姿态估计精度为0.0013deg/m）。从运行时间和配置来看算例依旧较大，无法完成嵌入式平台的实时定位，因此一种面向复杂情况的轻量级优化的雷达里程计LEGO-LOAM在2018年提出。

Additional information used by the methods

- Stereo: Method uses left and right (stereo) images
- Laser Points: Method uses point clouds from Velodyne laser scanner
- Loop Closure Detection: This method is a SLAM method that detects loop closures
- Additional training data: Use of additional data sources for training (see details)

	Method	Setting	Code	Translation	Rotation	Runtime	Environment	Compare
1	V-LOAM			**0.55 %**	**0.0013 [deg/m]**	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
J. Zhang and S. Singh: Visual-lidar Odometry and Mapping: Low drift, Robust, and Fast. IEEE International Conference on Robotics and Automation(ICRA) 2015.								
2	LOAM			0.57 %	0.0013 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
J. Zhang and S. Singh: LOAM: Lidar Odometry and Mapping in Real- time. Robotics: Science and Systems Conference (RSS) 2014.								
3	IMLS-SLAM++			0.61 %	0.0014 [deg/m]	1.3 s	1 core @ >3.5 Ghz (C/C++)	
4	SOFT2			0.65 %	0.0014 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
I. Cvišić, J. Česić, I. Marković and I. Petrović: SOFT-SLAM: Computationally Efficient Stereo Visual SLAM for Autonomous UAVs. Journal of Field Robotics 2017.								
5	IMLS-SLAM			0.69 %	0.0018 [deg/m]	1.25 s	1 core @ >3.5 Ghz (C/C++)	
J. Deschaud: IMLS-SLAM: Scan-to-Model Matching Based on 3D Data. 2018 IEEE International Conference on Robotics and Automation (ICRA) 2018.								

Figure 1 Odometry / SLAM Evaluation in KITTI

LEGO-LOAM将不同类型的特征点进行标注，并在匹配过程中仅在相同标签下进行匹配，同时运用两步骤LM方法将平面与边缘特征分开估计运动参数，从而实现使用较少的计算量达到与LOAM相近的性能。因此本文就LEGO_LOAM的算法思路和实施方案进行详细阐述，并给结合实际给出了部分优化方案及可行性分析。

二、算法流程

本部分参考"泡泡图灵智库"。

如Figure 2所示，系统由5个部分组成：首先是点云分割，将点云投影到一个范围图像中去分割。接下来是特征提取，然后雷达里程计估计位姿，特征会被处理后送到建图模块中，将它们配准到全局的点云地图中。最后，位姿集成模块使用位姿估计的结果和建图的结果获得最后的输出。

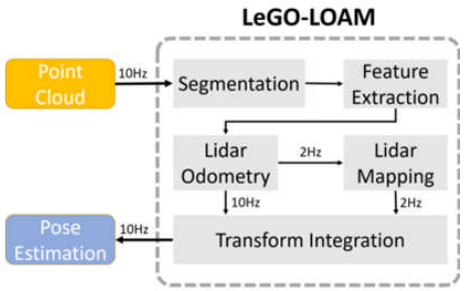

Figure 2 Pipeline of LEGO-LOAM

1、点云分割

使用了《Fast Segmentation of 3D Point Clouds for Ground Vehicles》中的方法进行地面点的区分，即靠下的7线激光雷达数据两两间通过角度判断是否属于平面，标记为地面的点不用于后续分割。

使用《Fast Range Image-based Segmentation of Sparse 3D Laser Scans for Online Operation》的方法在特征点所在的列和行进行索引从而聚类出组并打上标签，若组内点过少（<30）就会把这部分点视为不稳定的点给剔除掉（如树叶）。

处理后的点云如下图b中所示，其中红色点为地面点，其余点为不同组的分割点，将这些点存入Range Image中。

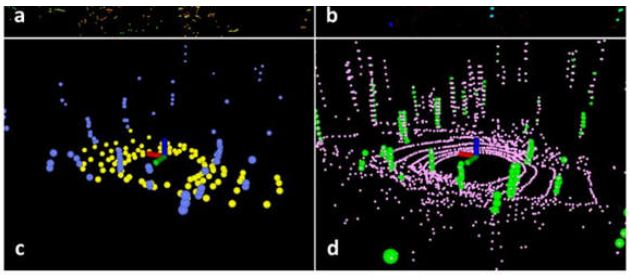

Figure 3 Feature extraction process for a scan in nosiy enviroment

2、特征提取

特征提取方法与LOAM中的方法类似，但是相比于直接从原始点云数据中提取特征点，LEGO-LOAM从已经分组的数据中提取特征，令S为以每个Range Image中同一行的连续点的点集，则的粗糙度可以表示为：

$$c = \frac{1}{|S| \cdot \|r_i\|} \left\| \sum_{j \in S, j \neq i} (r_j - r_i) \right\|$$

为了从所有方向均匀地提取特征，将Range Image水平划分为几个相等的sub-image，按每个点的粗糙度值对sub-image的每一行中的点进行排序，使用阈值来区分不同类型的特征，若$c < c_{th}$为边特征，若则$c > c_{th}$为平面特征。每次选取最小的特定数量的特征点用于建图，并通过降采样处理出更少的一部特征点用于后续匹配。如下所示，Figure 4为特征提取的示意图。

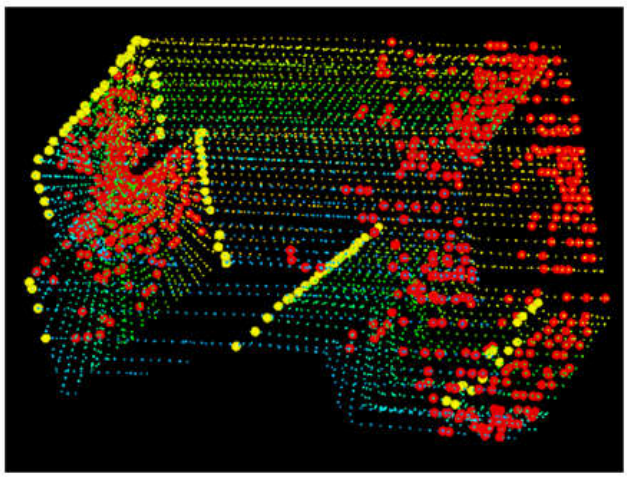

Figure 4 An example of extracted edge points (yellow) and planar points (red)

3、雷达里程计

雷达里程计模块估计连续两次扫描的位姿信息，使用从点-点和面-面匹配获得最终位姿变化信息，即要找到与上一帧特征数据同标签的对应点。这部分的内容可以从LOAM中找到。但是一些调整可以提升匹配的准确性与快速性：

- 标签匹配：在当前帧提取出的特征点，分别从上一帧具有相同标签的特征点降采样集合寻找对应点，以达到提高匹配准确性，缩小潜在对应特征数量的目的。
- 两步的优化：两步L-M优化方法，最佳转换T分两步找到。平面点匹配得到，边缘点匹配融合一步匹配结果得到，由此可以将计算量降低约35%。算法逻辑如Figure 5所示。

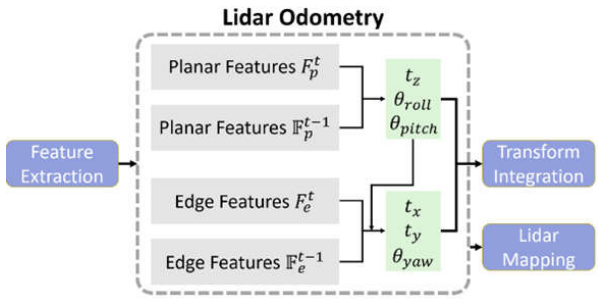

Figure 5 Two-step optimization for the lidar odometry'

4、雷达建图

雷达建图以较低的频率运行，将当前的特征与周围点云地图进行匹配修正位姿估计的结果，这里再次使用了LM方法。还是可以从LOAM中查看细节。LEGO-LOAM最大的不同就是最后的点云地图怎么存储，相比于存点云地图，它将每一帧的特征存储。然后从特征去生成周围的点云就有了两种方法，第一种方法，可以从可以将当前场景下可以看到的特征选取，简单点可以选择当前位置周围100m位置看到的特征。第二种方法就是pose-graph的方法，用局部共视的图去添加新的约束，这种方法可以利用Loop Closure进一步消除之前的累计误差。

三、代码框架

LEGO-LOAM的源代码依赖于PCL、OpenCV、GTSAM三个库。其中GTSAM是一个在机器人领域和计算机视觉领域用于平滑（smoothing）和建图（mapping）的C++库。它与g2o不同的是，g2o采用稀疏矩阵的方式求解一个非线性优化问题，而GTSAM是采用因子图（factor graphs）和贝叶斯网络（Bayes networks）的方式最大化后验概率。

与之前说介绍的算法内容一致，整个工程共分为imageProjection、featureAssociation、mapOptmization和transformFusion四个可执行文件。点云结构、传感器类型及算法精度等定义全部在件utility.h文件中完成。整体代码架构如下图所示：

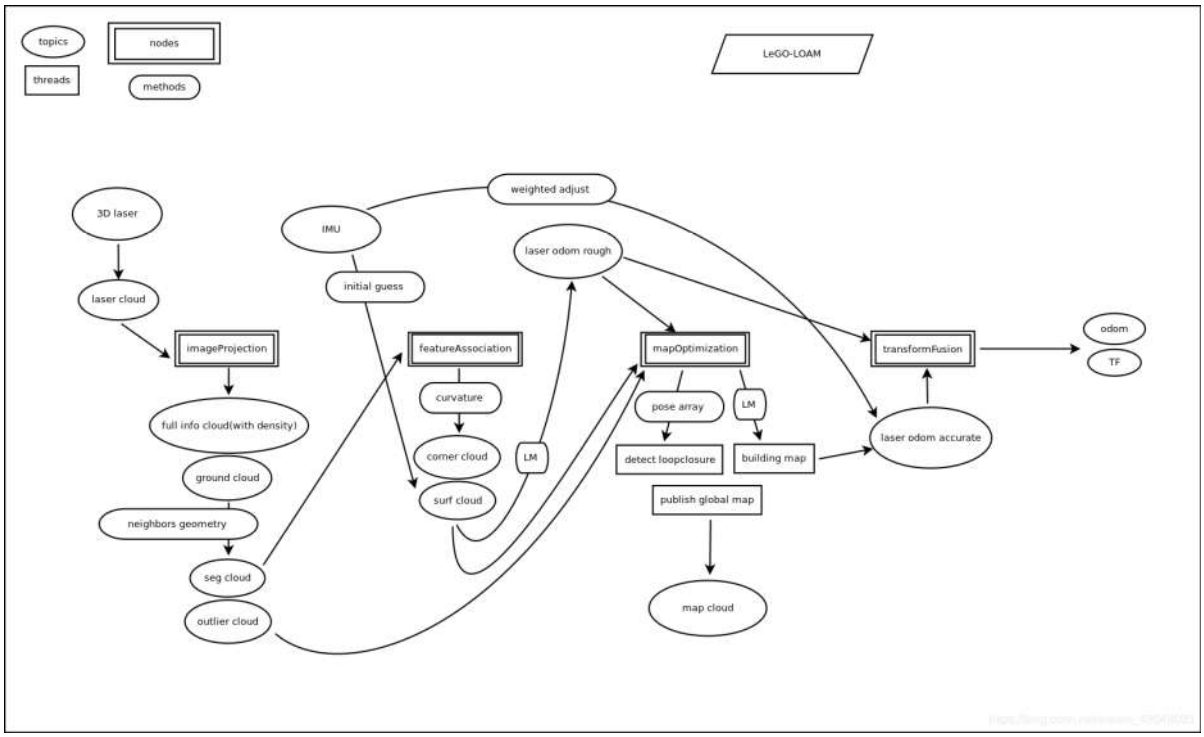

Figure 6 Code Architecture

四、离线测试

- 测试硬件：VLP-16。它具有30°（15°）的垂直视场（FOV）和360°的水平FOV。16通道传感器提供2°的垂直角分辨率。水平角分辨根据旋转速率不同，从0.1°到0.4°变化。测试时使用10Hz的扫描速率，其提供0.2°的水平角分辨率。
- 测试环境：测试场地位于公司地下停车场，场地范围约100×60m^2,且测试车速约为30km/h,测试路径存在闭环。
- 电脑配置：Intel Core i5-9600K CPU 3.7GHZ×6，Ubuntu 16.04。

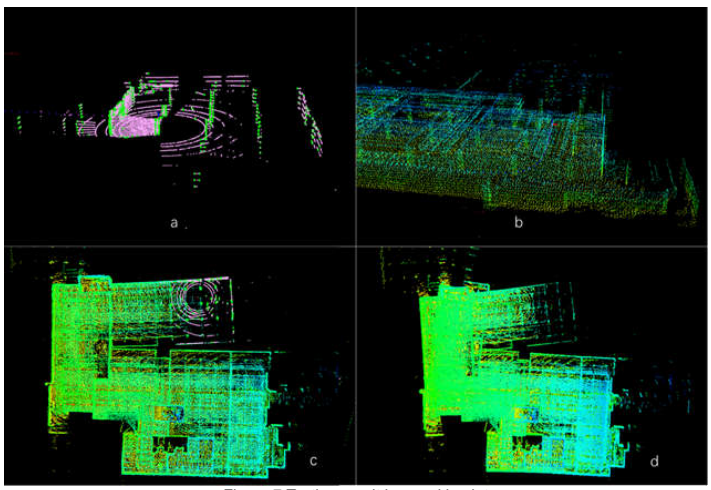

Figure 7 Testing result in a parking lot

上图a给出了单帧的边特征和面特征（绿色和红色），图b给出了局部地图，图c给出了增加了回环检测后的实时测试效果，图d给出了未增加回环的实测效果图。从图c和图d中可以看出，在进行直线行进过程中，所建地图边界点无毛刺或重合，测试精度较高，而旋转过程中易出现歪斜，但由于测试空间较小，因而形成可以通过闭环检测消除累积误差。从图b可以看出LEGO-LOAM建立局部地图平滑性高，3D结构信息清楚，噪点数量较少，在一定程度上还原了测试场地的3维环境信息。由于所测试的数据集没有Ground-Truth,因而无法进行进一步的定量分析。

五、可行性分析

现有代码在地下停车场这一小场景、存在闭环、障碍物较少的数据集上有良好的表现，定位精度粗略估计可以达到20cm以内。但是单看前端的雷达里程计的定位精度仍旧不高。因此在开放的室外场景下、高速运动场景、移动障碍物较多的场景下均无法保证可靠性，仍需结合现有的其他传感器进行融合处理。

从运行效率和所占内存上来看，在上述的电脑配置下，算法CPU占有率最高为150%，内存占有率最高为2.5%。通过其原始论文可以看到，计算资源占有量在可控范围内，应该可以完成实时的嵌入式平台计算。而所需内存随路径增长而增加，所需空间较大（400M），因而后续对地图管理的工作也是难点之一。

由于现在的测量数据均是VLP-16的测量结果，数据源精度非常高，后期如果换成其他的雷达数据，可能需要进行大量的数据标定，数据滤波，传感器融合的工作，现目前无法评估在其他类型传感器数据集上本算法的表现。

综上，如若将应用场景定位在小场景且干扰较小的情况下，如地下停车库、隧道内等，本算法具有实施的可能性。通过优化前端雷达里程计精度及传感器融合处理，在传感器精度可以保证的情况下具有较强的实践意义。但由于长时间的无闭环状态使得算法无法消除累积误差，高速情况下场景重复且点云匹配难度较大，因此不建议将本算法应用于高动态、长距离、强干扰的场景下。

六、技术难点

1、特征点选型

前端现有LEGO-LOAM代码中选择地面点与边缘点作为标签特征进行匹配，但从测试结果看，所选特征在旋转状态时表现不佳。初步分析是由于旋转过程中边缘特征丢失或激光点漂移造成的。此外，车辆行驶环境复杂，不同环境对特征点要求也不尽相同。因此如何妥善优化，选择更加适配的特征点是后续工作的难点之一。

2、定位精度优化

前端定位需要尽可能的提升以满足各种场景的需求，特别是在有动态障碍物的场景下仍需保证较强的定位精度，这就需要针对现有代码进行测试分析，定位到问题点从而进一步进行优化提升。

3、多传感器融合

如果前端精度无法达到需求，可以考虑将里程计和IMU信息进行融合，达到融合定位的目的。

4、地图存储

地图是回环检测、局部优化、全局优化的必要基础。因此如何在保证算法精度与稳定性的情况下合理降低存储地图（主要为关键帧）所占内存以满足嵌入式移植标准是后续工作必须解决的另一问题。

5、回环检测

现有代码在实际运行中回环检测效果依赖于几何信息，而这种几何信息主要依赖于前端里程计的定位精度，因而如何保证回环检测的稳定性是后续工作的主要目标之一

6、依赖库移植

LEGO-LOAM代码依赖于PCL、OPENCV和GTSAM，其中GTSAM是基于因子图的解决方案，现目前在嵌入式平台中的移植并不多见，移植过程的难度现无法预估。此外，这些依赖库在过程中需要大量的代码裁剪，环境编译，版本匹配等工作，工作量较大。

七、工作量预算

前端优化定位根据项目难点与关键点，下表将后续待完成工作做拆分整理，并给出初步的工作量预算。需要说明的是下面所列工作之间存在互相交叠，给出的工作量预估是完整完成该项工作需要的粗略估计时间。

Table 1 Contents & Workload Estimation

工作内容	工作量估计（1人/月）
前端精度定位与优化	1
特征点选型	1
多传感器融合	1
后端可靠性优化	1
关键帧选择与地图存储优化	1
工程化细节及算法优化	2
代码移植	1
迭代测试与优化	4

八、总结

本文在阐述现有激光雷达SLAM方法的基础上，从算法原理、代码框架、仿真测试三方面全面介绍了LEGO-LOAM的内容，同时结合硬件配置给出了这套方法在实际工程应用中的可行性及难点，最后拆分出了待完成工作并预估了完成各项工作所需时间。

参考目录

论文概要：https://mp.weixin.qq.com/s/DxaleLWAx4BA2O6dlLeYfQ

代码解读：https://blog.csdn.net/weixin_42048023/article/details/87624650

论文翻译：https://blog.csdn.net/wykxwyc/article/details/89605721

无标签