

# 目录

1 传感器知识	1.1 <a href="#">基本原理</a>	
	1.2 <a href="#">传感器分类</a>	
	1.3 <a href="#">主流厂家及设备介绍</a>	国内外厂家，设备参数性能
	1.4 <a href="#">三维激光雷达应用介绍</a>	
2 三维激光 SLAM 研究现状	2.1 数据预处理	<a href="#">运动畸变去除</a> <a href="#">点云分割、特征提取</a>
	2.2 前端扫描匹配	<a href="#">迭代最临近点及变种 ICP</a> <a href="#">相关性扫描匹配 CSM</a> <a href="#">正态分布变换 NDT</a> <a href="#">其他</a>
	2.3 后端优化	<a href="#">基于滤波器</a> <a href="#">基于图优化</a>
	2.4 闭环检测	<a href="#">基于几何距离</a> <a href="#">基于点云形状</a>
	2.5 <a href="#">重点与难点</a>	
3 LeGO-LOAM 开源代码详细讲解	3.0 <a href="#">基于 LOAM 开发的代码</a>	
	3.1 <a href="#">代码结构</a>	
	3.2 基础理论	3.2.1 <a href="#">点云分割与特征提取</a> 3.2.2 <a href="#">激光雷达里程计</a> 3.2.3 <a href="#">地图优化</a> 3.3.4 <a href="#">回环检测</a>

# 1.传感器知识

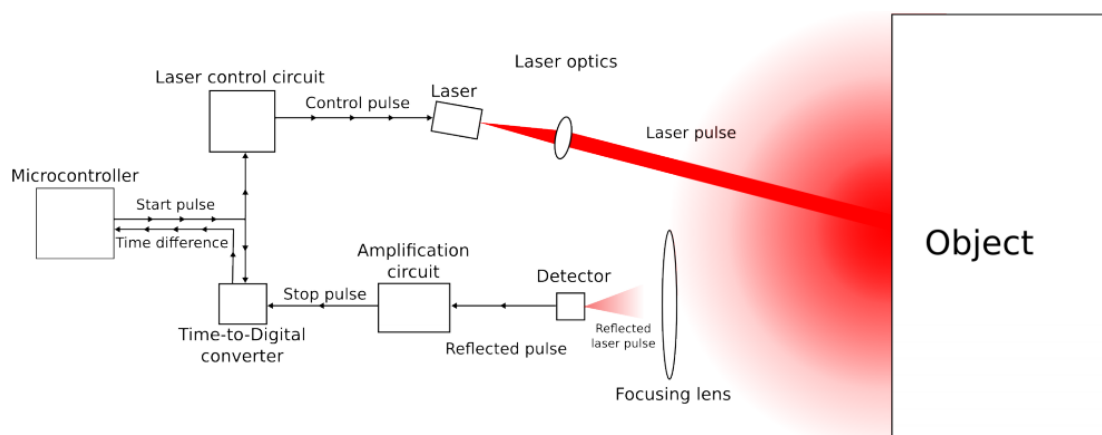
## 1.1 基本原理

激光雷达是一项主动式的现代光学测量技术，将传统雷达技术与激光技术相结合形成的新技术。由于激光具有高亮度性、高方向性、高单色性和高相干性等特点，因此激光雷达具有一系列独特的优点：角分辨率高、距离分辨率高、速度分辨率高、测速范围广、能获得目标的多种图像、抗干扰能力强。激光雷达的传统应用领域有军事、遥感、测绘、安防、环境探测等，但随着自动驾驶技术的火爆，其公认的最有前景的应用是无人车的三维环境传感和高精地图构建。

激光雷达工作的基本原理：飞行时间测量法（Time of Flight）

- 1、激光雷达中的激光器发射出一束超短激光脉冲；
- 2、激光投射到物体上后发生漫反射，激光接收器接收漫反射光；
- 3、通过激光光束在空中的飞行时间，准确计算得出目标物体到传感器间的距离。

通过测量反射、散射回波信号的时间间隔、频率变化、波束所指方向等就可以确定目标的**距离、方位和速度**等信息，然后结合激光器本身的位置信息和姿态角度信息，准确计算出目标表面回波点的三维坐标。若激光束不断地扫描目标物，就可以得到目标物上全部目标点的数据，用此数据进行成像处理后，就可得到精确的三维立体图像。至于目标的径向速度，可以由反射光的多普勒频移来确定。



激光雷达在一些关键技术指标上远远超越了其他遥感探测技术，使其在很多领域得以广泛应用。具体包括：

- （1）数据密度大

激光波束窄，探测次数多，因而采集数据量更大。每秒可测量数十万个点，对真实物体表面(如地面)的还原和建模带来极大方便。同时，可调节节点采集间隔，大大提高了适用性和工作效率。

#### （2）数据精度高

由于激光波长短、频率高，可以使激光雷达达到极高的测量精度。高精度测量激光雷达测量精度可达毫米以下，机载激光雷达测量精度也可达厘米级。

#### （3）植被穿透能力强

激光在植被中传播时，可以在树冠、树枝、地面等多个高程发生反射，从而得到多次回波数，这是其他雷达所不具备的优势。特别是得到的地面回波数据，有效克服了植被影响，使精确探测地面真实地形成为可能。

#### （4）不受太阳高度角和阴影影响

激光雷达为主动测量式雷达，不依赖自然光，因而与传统方式相比，其获取数据的精度不受时间、太阳高度角和地物阴影的限制和影响，可以二十四小时全地形作业。

#### （5）隐蔽性好、抗干扰能力强

激光传播方向性好、波束窄，只在传播路径上存在，难以发现和截获。同时，激光雷达口径小，且定向接收，只接收指向区域回波，接收干扰信号的概率极低。

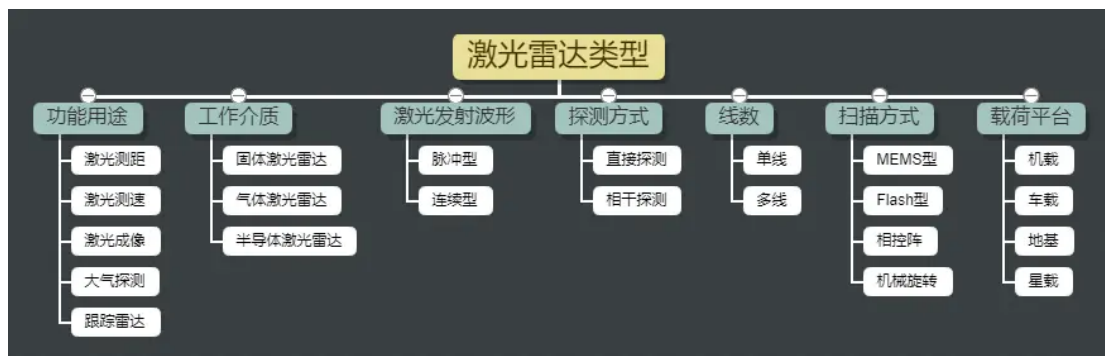
#### （6）体积小、重量轻、作业效率高

激光雷达发射口径只有几厘米，重量小的可以单人手持使用，相较其他雷达设备要轻便、灵巧得多，不但可以大量节约人力、物力资源，而且可以使工作变得更加简单快捷，可应用的领域也更广。

#### 激光雷达的缺点

首先，工作时受天气和大气影响大。激光一般在晴朗的天气里衰减较小，传播距离较远。而在大雨、浓烟、浓雾等坏天气里，衰减急剧加大，传播距离大受影响。如工作波长为  $10.6\ \mu\text{m}$  的  $\text{CO}_2$  激光，是所有激光中大气传输性能较好的，在坏天气的衰减是晴天的 6 倍。地面或低空使用的  $\text{CO}_2$  激光雷达的作用距离，晴天为 10—20km，而坏天气则降至 1 km 以内。而且，大气环流还会使激光光束发生畸变、抖动，直接影响激光雷达的测量精度。

其次，由于激光雷达的波束极窄，在空间搜索目标非常困难，直接影响对非合作目标的截获概率和探测效率，只能在较小的范围内搜索、捕获目标，因而激光雷达较少单独直接应用于战场进行目标探测和搜索。



## 1.2 传感器分类

激光雷达是激光、大气光学、雷达、光机电一体化和电算等技术相结合的产物，几乎涉及了物理学的各领域。目前激光雷达种类繁多，可以按照所用激光器的探测技术、功能用途、运载平台进行分类。

**按工作介质分类：**工作介质表示产生激光的激励介质是什么，气体激光器利用气体作为工作物质产生激光的器件，固体激光器是用固体激光材料作为工作物质的激光器，近年来，激光雷达发展的重点是二极管泵浦固体激光雷达。

### **按激光发射波形分类：**

连续型激光雷达从激光的原理来看，连续激光就是一直有光出来，就像打开手电筒的开关，它的光会一直亮着（特殊情况除外）。连续激光是依靠持续亮光到待测高度，进行某个高度下数据采集。

脉冲型激光雷达输出的激光是不连续的，而是一闪一闪的。脉冲激光的原理是发射几万个的激光粒子，根据国际通用的多普勒原理，从这几万个激光粒子的反射情况来综合评价某个高度的风况。

### **按探测方式分类：**

直接探测激光雷达的基本结构与激光测距机颇为相近。工作时，由发射系统发送一个信号，经目标反射后被接收系统收集，通过测量激光信号往返传播的时间而确定目标的距离。至于目标的径向速度，则可以由反射光的多普勒频移来确定，也可以测量两个或多个距离，并计算其变化率而求得速度。

相干探测型激光雷达有单稳与双稳之分，在所谓单稳系统中，发送与接收信号共用一个光学孔径，并由发送-接收开关隔离。而双稳系统则包括两个光学孔径，分别供发送与接收信号使用，发送-接收开关自然不再需要，其余部分与单稳系统相同。

### 按线数分类：

单线激光雷达主要用于规避障碍物，其扫描速度快、分辨率强、可靠性高，在角频率和灵敏度反映快捷，测试障碍物的距离和精度上都比多线准确精确。但是，单线雷达只能平面式扫描，不能测量物体高度，当前主要应用于服务机器人身上，如我们常见的扫地机器人。

多线激光雷达主要应用于汽车的雷达成像，相比单线激光雷达在维度提升和场景还原上有了质的改变，可以识别物体的高度信息。目前在国际市场上推出的主要有 4 线、8 线、16 线、32 线和 64 线。但价格高昂。其细分可分为 2.5D 激光雷达及 3D 激光雷达。2.5D 激光雷达与 3D 激光雷达最大的区别在于激光雷达垂直视野的范围，前者垂直视野范围一般不超过  $10^{\circ}$ ，而后者可达到  $30^{\circ}$  甚至  $40^{\circ}$  以上，这也就导致两者对于激光雷达在汽车上的安装位置要求有所不同。

### 按扫描方式分类：

机械激光雷达带有控制激光发射角度的旋转部件，而固态激光雷达则依靠电子部件来控制激光发射角度，无需机械旋转部件。

机械激光雷达由光电二极管、MEMS 反射镜、激光发射接受装置等组成，其中机械旋转部件是指可  $360^{\circ}$  控制激光发射角度的 MEMS 发射镜。

固态激光雷达与机械雷达不同，它通过光学相控阵列、光子集成电路以及远场辐射方向图等电子部件代替机械旋转部件实现发射激光角度的调整。固态激光雷达常见的技术实现方式有三类：微机电（MEMS）、相控阵（OPA）和泛光面阵式（3D）。泛光面阵式激光雷达是目前全固态激光雷达中最主流的技术。Flash 型激光雷达能快速记录整个场景，避免了扫描过程中目标或激光雷达移动带来的各种麻烦，它运行起来比较像摄像头。相控阵激光雷达搭载的一排发射器可以通过调整信号的相对相位来改变激光束的发射方向。

### 补充：

Quanergy 在 2016 年公开固态激光雷达 S3 的工作原理，可以看出 S3 采用的是光学相控阵技术实现激光扫描，与传统机械扫描技术相比，光学相控阵扫描技术有三大优势：

- 扫描速度快：光学相控阵的扫描速度取决于所用材料的电子学特性和器件的结构，一般都可以达到 MHz 量级以上。
- 扫描精度或指向精度高：光学相控阵的扫描精度取决于控制电信号的精度（一般为电压信号），可以做到  $\mu\text{rad}$ （千分之一度）量级以上。
- 可控性好：光学相控阵的光束指向完全由电信号控制，在允许的角度范围内可以做到任意指向，可以在感兴趣的目标区域进行高密度的扫描，在其他区域进行稀疏扫描，这对于

自动驾驶环境感知非常有用。

**但光学相控阵扫描技术也有它的缺点：**

- 易形成旁瓣，影响光束作用距离和角分辨率：光束经过光学相控阵器件后的光束合成实际是光波的相互干涉形成的，干涉效果易形成如下图所示的旁瓣，使得激光能量被分散。
- 加工难度高：光学相控阵要求阵列单元尺寸必须不大于半个波长，一般目前激光雷达的工作波长均在 1 微米左右，这就意味着阵列单元的尺寸必须不大于 500 纳米。而且阵列数越多，阵列单元的尺寸越小，能量约往主瓣集中，这就对加工精度要求更高。

Velodyne 从 VLP-16 产品面世后才开始宣称这款 16 线激光雷达采用“固态混合”，Velodyne 所有的产品在俯仰方向（垂直于水平面方向）均采用了电子扫描技术，在方位方向（水平方向）采用机械 360° 旋转扫描。

**Velodyne 的这种固态扫描技术具有以下优点：**

- 扫描速度快：扫描速度只决于发射模块的电子学响应速度，不受材料的特性影响，可以实现比光学相控阵更高的扫描频率。但其扫描角度一定设计好后就完全固定，不能通过电控进行改变。
- 接收视场小：这种扫描技术是一种发射和接收同步扫描技术，接收视场小，抗光干扰能力强，信噪比高。
- 可承受高的激光功率：这种扫描技术完全是在自由空间中进行，可以采用高峰值功率的激光脉冲进行高信噪比的探测。

**同时，这种扫描技术也存在以下问题：**

- 实现二维扫描比较困难：按照目前这种非集成式的模块化设计难以实现二维扫描，必须通过机械或其他方式实现另一维的扫描。集成化是这种技术发展的必然趋势，也是实现二维扫描的关键。
- 扫描角度固定：但其扫描角度一定设计好后就完全固定，不能通过电控进行改变。
- 装调工作量大：需要将发射和接收模块进行精密光学对准装配，工作繁复，工作量大，大批量生产难度大。

**按载荷平台分类：**

机载激光雷达是将激光测距设备、GNSS 设备和 INS 等设备紧密集成，以飞行平台为载体，通过对地面进行扫描，记录目标的姿态、位置和反射强度等信息，获取地表的三维信息，

并深加工得到所需空间信息的技术。在军民用领域都有广泛的潜力和前景。机载激光雷达探测距离近，激光在大气中传输时，能量受大气影响而衰减，激光雷达的作用距离在 20 千米以内，尤其在恶劣气候条件下，比如浓雾、大雨和烟、尘，作用距离会大大缩短，难以有效工作。大气湍流也会不同程度上降低激光雷达的测量精度。

车载激光雷达又称车载三维激光扫描仪，是一种移动型三维激光扫描系统，可以通过发射和接受激光束，分析激光遇到目标对象后的折返时间，计算出目标对象与车的相对距离，并利用收集的目标对象表面大量的密集点的三维坐标、反射率（激光打到不同的射物上，能反射回来的激光占发射出去的激光比例）等信息，快速复建出目标的三维模型及各种图件数据，建立三维点云图，绘制出环境地图，以达到环境感知的目的。车载激光雷达在自动驾驶“造车”大潮中扮演的角色正越来越重要，诸如谷歌、百度、宝马、博世、德尔福等企业，都在其自动驾驶系统中使用了激光雷达，带动车载激光雷达产业迅速扩大。

地基激光雷达可以获取林区的 3D 点云信息，利用点云信息提取单木位置和树高，它不仅节省了人力和物力，还提高了提取的精度，具有其它遥感方式所无法比拟的优势。通过对国内外该技术林业应用的分析和对该发明研究后期的结果验证，未来将会在更大的研究区域利用该技术提取各种森林参数。

星载激光雷达采用卫星平台，运行轨道高、观测视野广，可以触及世界的每一个角落。为境外地区三维控制点和数字地面模型的获取提供了新的途径，无论对于国防或是科学研究都具有十分重大意义。星载激光雷达还具有观察整个天体的能力，美国进行的月球和火星等探测计划中都包含了星载激光雷达，其所提供的数据资料可用于制作天体的综合三维地形图。此外，星载激光雷达载植被垂直分布测量、海面高度测量、云层和气溶胶垂直分布测量以及特殊气候现象监测等方面也可以发挥重要作用。

### 1.3 主流厂家及设备介绍

全球激光雷达主流厂商仍然以外企为主，国内比较有影响力的激光雷达生产企业有速腾聚创、北醒光子、镭神智能、北科天绘、禾赛科技、光珀智能等

厂商	国别	成立	主要投资机构
Velodyne	美国	1983	百度、福特
Quanergy	美国	2012	三星、德州仪器、马斯克、戴姆勒、德尔福
速腾聚创	中国	2014	东方富海、复星锐正、北汽产投

北醒光子	中国	2015	IDG、沃勒斯机器人、顺为、凯辉汽车基金
禾赛科技	中国	2013	远瞻资本、磐谷创投、百度、高达投资、将门创投
北科天绘	中国	2005	联想之星、云辉资本、Star vc
光珀智能	中国	2013	浙江金控
LeddarTech	加拿大	2007	BDC、Venture captical、欧司朗、德尔福
IBEO	德国	2009	采埃孚
Luminar	美国	2012	1517fund、GVA captical
Innoviz	以色列	2016	Zohar Zisapel、三星、软银
Cepton	美国	2016	
Innovusion	美国	2016	
Orix Vision	以色列	2009	Bessemer Venture Partner
Tetra Vue	美国	2008	Nautilars、三星、Robert Venture、富士康
镭神智能	中国	2015	招商、北极光、达晨、如山
Ouster	美国	2015	Cox Enterprises
Strobe	美国	2014	通用

下面是一些主流三维车载激光雷达的性能参数：

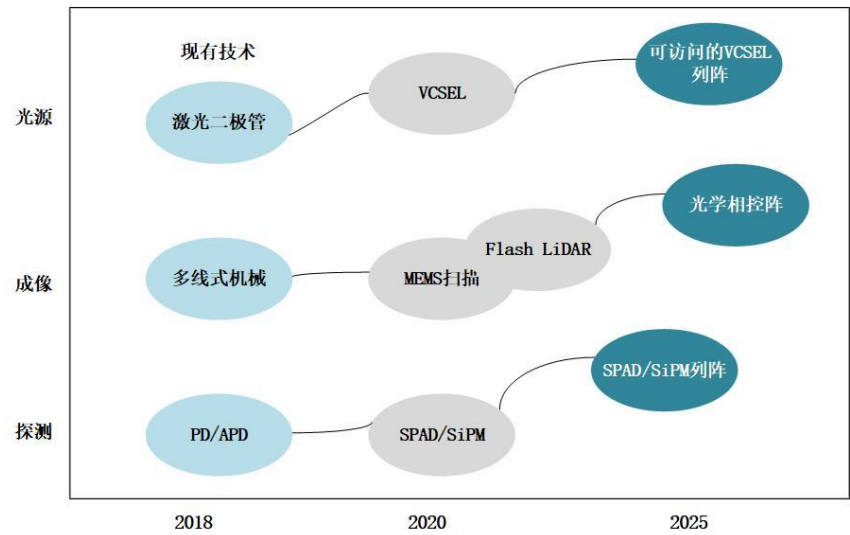
厂家	型号	规格
Velodyne	HDL-64E	线束：64 线 水平视角/分辨率：360° /0.17° at 600 rpm 垂直视角（仰角）/分辨率：+2° , -24.8° /0.33° (upper 32 lasers) 0.5° (lower 32 lasers) 测量精度：±2cm 视角更新率：5---20Hz 测距范围：up to 120m
	HDL-32E	线束：32 线 水平视角/分辨率：360° / 0.16° at 600 rpm 垂直视角（仰角）/分辨率：+10° , -30° / 1.33° 测量精度：±2cm



		视角更新率：5---20Hz 测距范围：up to 100m
	VLP-16	线束：16 线 水平视角/分辨率：360° / 0.2° at 600 rpm 垂直视角（仰角）/分辨率：+15° , -15° / 2° 测量精度：±3cm 视角更新率：5---20Hz 测距范围：up to 100m
禾赛科技	Pandora40	线束：40 线 水平视角/分辨率：360° / 0.2° (10Hz), 0.4°(20Hz) 垂直视角（仰角）/分辨率：+7° , -16° / 0.33°(-6° ~+2° ), 1° (-16° ~-6° ,+2° ~+7° ) 测量精度：±5 cm (0.3 m~0.5 m); ±2 cm (0.5 m~200 m) 视角更新率：10Hz, 20Hz 测距范围：0.3 m~200 m (20% 反射率)
速腾聚创	RS-LiDAR-16	线束：16 线 水平视角/分辨率：360° / 0.09° (5Hz) , 0.36° (20Hz) 垂直视角（仰角）/分辨率：+15° , -15° / 2° 测量精度：±2 cm 视角更新率：5/10/20Hz 测距范围：20cm 至 150 米（目标反射率 20%）
镭神	C16	线束：16 线 水平视角/分辨率：360° / 0.09° (5Hz) , 0.36° (20Hz) 垂直视角（仰角）/分辨率：+15° , -15° / 2° 测量精度：±3 cm 视角更新率：5/10/20Hz 测距范围：50cm 至 150 米（目标反射率 70%）

目前车载激光雷达处于起步阶段，行业呈现多技术路线并存的状态，随着时间的推移，基于 MEMS 的扫描激光雷达最早将在 2020 年开始部署，而纯固态激光雷达技术的成熟也将

成为 MEMS 技术的一大挑战，制造工艺、分辨率、成本也成为了每种技术成功的关键，发射极类型、探测技术以及光束转向技术都是行业技术的难点，下图是车载激光雷达的技术发展路线图：



### 1.4 三维激光雷达应用介绍

激光雷达在民用领域主要集中在一些智能设备上，如 AGV 小车、扫地机器人和自动驾驶领域，如表所示：

装备类型	激光引导 AGV	扫地机器人	自动驾驶/建模
图片			
激光雷达类型	8/16 线激光雷达为主	单线激光雷达为主	多线激光雷达
应用领域	工业、物流	清洁	无人驾驶
简述	激光引导 AGV 是普通 AGV 的进一步升级产品，体现出比传统 AGV 更智能化的特点。行业有一定的价格敏感性。目前国内厂商多采用进口产品，价格较贵，进口替代市场有一定机会。	扫地机器人是激光雷达技术家用化的典型代表。但与一般的小家电产品相同，该行业价格敏感度很高，加之单线雷达技术门槛较低，因此普遍售价在 300 元左右。	目前已自动驾驶系统训练为主要应用，测试市场为主的情况下，对产品的标准化程度要求不高，耐用性和可靠性要求也不高，价格敏感度低，目前全球市场主要被 VELOCITYNE 主导。

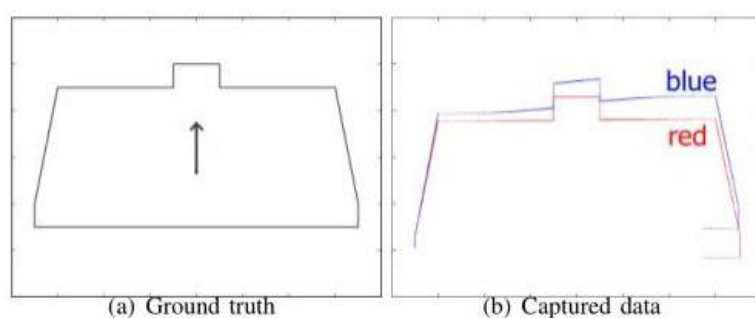
## 2.三维激光 SLAM 研究现状

### 2.1 数据预处理

#### 2.1.1 运动畸变去除

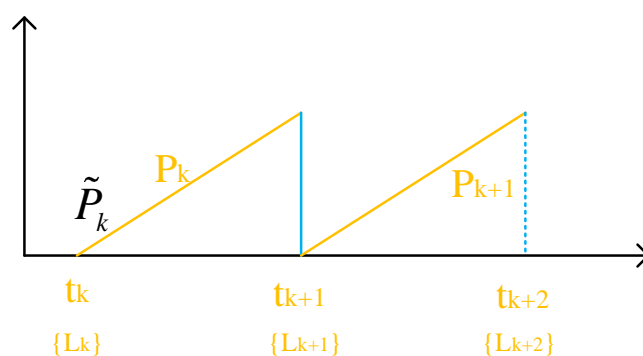
去除畸变的方法：1、基于纯激光雷达的 2、基于里程计辅助的

产生原因：常用激光雷达是旋转扫描工作方式，每个激光点打出去的时间点不同，考虑到无人车行进过程中每一帧点云的帧头和帧尾之间存在一段运动量，也就是说一帧内的激光点并不是处于同一个局部坐标系。当激光雷达一帧扫描周期为 0.1s，且车体运动速度为 20km/h 的情况下，每帧点云第一个激光点和最后一个激光点所处的坐标系之间偏差达到 55.6cm，可见原始点云以及提取的特征点均需要进行去畸变过程才可以用于帧间配准。点云的去畸变处理需要预先知道帧间运动量，而求解帧间运动量又依赖于去畸变后特征点的匹配，因此我们可以用位姿插值的办法将两个操作放在一个算法中同时处理。



运动畸变示意图 [sdn.net/qj\\_38338228](https://www.cnblogs.com/qj_38338228)

- 基于纯激光雷达的运动畸变去除



$P_k$  : 第 k 帧期间扫描得到的一帧点云;  $\tilde{P}_k$  : 将  $P_k$  里的每个点投影到时刻  $t_k$  对应坐标

系下；设帧头时刻对应的局部坐标系为 $\{L_k\}$ ，帧尾时刻对应的局部坐标系为 $\{L_{k+1}\}$ ，在第  $k$  帧期间，车体的运动量等价于 $\{L_{k+1}\}$ 相对于 $\{L_k\}$ 的旋转和平移，用 $\mathbf{TR}_k$ 表示， $\mathbf{TR}_k = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z]^T$ 。激光雷达在实际工作中采用的扫描频率为 10Hz，即一帧点云的扫描周期为 0.1s，假设在扫描周期内无人车的运动是平滑的，基于匀速运动模型，在时刻  $t_i$  的激光点  $X_{(k,i)}^L$ ，其相对 $\{L_k\}$ 的运动量为  $s \times \mathbf{TR}_k$ ， $s$  表示某点在一帧扫描内的相对时间。

假设 $\mathbf{TR}_k$ 已知，则可以将 $X_{(k,i)}^L$ 投影至 $\{L_k\}$ ：

$$\tilde{X}_{(k,i)}^L = R_{(k,i)}^{-1} (X_{(k,i)}^L - T_{(k,i)})$$

其中位移矩阵 $T_{(k,i)} = s \cdot [t_x, t_y, t_z]^T$ ，旋转矩阵 $R_{(k,i)} = R_{z(k,i)} R_{x(k,i)} R_{y(k,i)}$ ，LOAM 中定义的载体坐标系是 X 向左、Y 向上、Z 向前，旋转矩阵的旋转顺序是 **YXZ**，**转角顺时针为正**如下：

$$R_{y(k,i)} = \begin{bmatrix} \cos(s \cdot \theta_y) & 0 & \sin(s \cdot \theta_y) \\ 0 & 1 & 0 \\ -\sin(s \cdot \theta_y) & 0 & \cos(s \cdot \theta_y) \end{bmatrix}$$

$$R_{x(k,i)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(s \cdot \theta_x) & -\sin(s \cdot \theta_x) \\ 0 & \sin(s \cdot \theta_x) & \cos(s \cdot \theta_x) \end{bmatrix}$$

$$R_{z(k,i)} = \begin{bmatrix} \cos(s \cdot \theta_z) & -\sin(s \cdot \theta_z) & 0 \\ \sin(s \cdot \theta_z) & \cos(s \cdot \theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

通过上述的旋转和平移，就能把帧内所有的点云归于统一的局部坐标系下，实现去除畸变的目的。实际计算中， $\mathbf{TR}_k$ 是位姿估计的待估量，是未知的，所以点云畸变消除的这一步计算是和前端位姿估计一起进行的。

#### ● 基于里程计辅助的运动畸变去除

直接测量机器人的位移和角度，具有较高的局部角度测量精度，具有较高的局部位置测量精度。用 CPU 读取激光雷达数据，同时单片机上传里程计数据，两者进行时间同步，在 CPU 上统一进行运动畸变去除。

流程：

- 1、已知当前激光帧的起始时间  $t_s, t_e$ 。
- 2、两个激光束间的时间间隔  $\Delta t$
- 3、里程计数据按照时间顺序存储在一个队列里。
- 4、求解当前帧激光数据中的每一个激光点对应的里程计数据（即机器人位姿）
- 5、根据求解的位姿把所有的激光点转换到同一坐标系下
- 6、重新封装成一帧激光数据发布出去

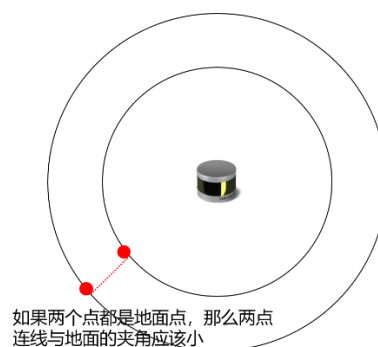
## 2.1.2 点云分割和特征提取

在城市道路环境下，激光雷达扫描获取的点云包含丰富的规整物体，如墙壁、地面、建筑物棱线、行道树、电线杆等。仔细观察这些规整物体，它们可以用两类基本几何结构加以抽象表示：平面和直线。平面即墙壁、地面等大面积的平整区域，直线则表示行道树树干、电线杆、建筑物棱线等呈一条直线分布的区域，城市道路环境下这些“线”和“面”通常较为均匀地分布在三维空间中的各个位置，并且它们的朝向（“线”的方向与“面”的法线方向）分布也较为均匀，因此非常有利于帧间点云的配准。我们提取点云中的“线”和“面”，并在相邻点云中分别寻找配对的“线”和“面”，进而估算帧间运动。

LeGO-LOAM 中的点云分割和特征提取方法：

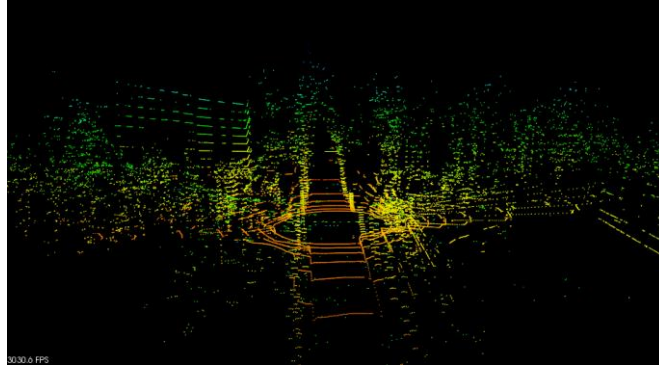
四个字，简单粗暴。点云分割的目的是分割出线、面特征点的候选点，尽可能减小特征提取时处理的点云数量，提高处理效率和特征提取精度。

对于面特征点，根据激光线束的视角范围，划分出能看到地面的线束，以 velodyne VLP-16 为例，假使下视的 7 根线束可以采集到地面点，在此基础上根据当前点和相邻扫描线、同一方位角的邻点的连线和水平面的夹角，判断当前点是否为面特征点的候选。



对于线特征点，计算当前点和上、下、左、右四个邻点的关系，左右邻点表示的是同一条扫描线、相邻方位角上的两点，上下邻点表示的是相邻扫描线、同一方位角上的相邻两点，

计算相邻两点间的距离和角度关系，足够接近的一群点云分组成一个簇，并被分配一个唯一的标签。假设一个机器人在嘈杂的环境中工作，小物体，例如树叶，可能会形成微不足道的不可靠的特征，因为在两次连续扫描中不太可能看到同一片树叶。为了利用分割后的点云进行快速可靠的特征提取，我们省略了不到 30 个点的聚类。经历了上述点云分割操作后，我们仅保留了大物体上（例如树干、电线杆）的点和地面点以供进一步处理。



如上图所示，可以发现在“面”上的单根扫描线呈现一条直线段分布，而“线”则把一根完整的扫描线分隔成朝向不同的两段，在“线”处形成一个转折点。定义激光点的局部平滑度：

$$S_i = \frac{1}{2n} \left\| \sum_{j \in [i-n, i+n], j \neq i} \frac{(P_i - P_j)}{\|P_i - P_j\|} \right\|$$

其中  $n$  表示选取与点  $P_i$  相同激光扫描线的邻近点的个数， $S_i$  表示的是点  $P_i$  与其邻近  $2n$  个激光点形成的单位向量之和的模。当扫描在“面”上时， $P_i$  与其邻近点分布在近似一条直线上，因此这些邻近点与  $P_i$  构成的向量相互抵消，使得  $S_i$  足够小。反之，当扫描在“线”上时，邻近点与  $P_i$  形成的向量之间无法相互抵消，因此使得  $S_i$  较大。因此设定面特征阈值  $Ths_{plane}$  和线特征阈值  $Ths_{edge}$ ，若  $S_i < Ths_{plane}$ ，则认为  $P_i$  为面特征点，若满足  $S_i > Ths_{edge}$ ，则认为  $P_i$  为线特征点。此外考虑到均匀分布的特征点更有利于帧间配准，因此我们在提取线特征点和面特征点时应分区域提取，使得每一块区域内特征点的数量大致相当。

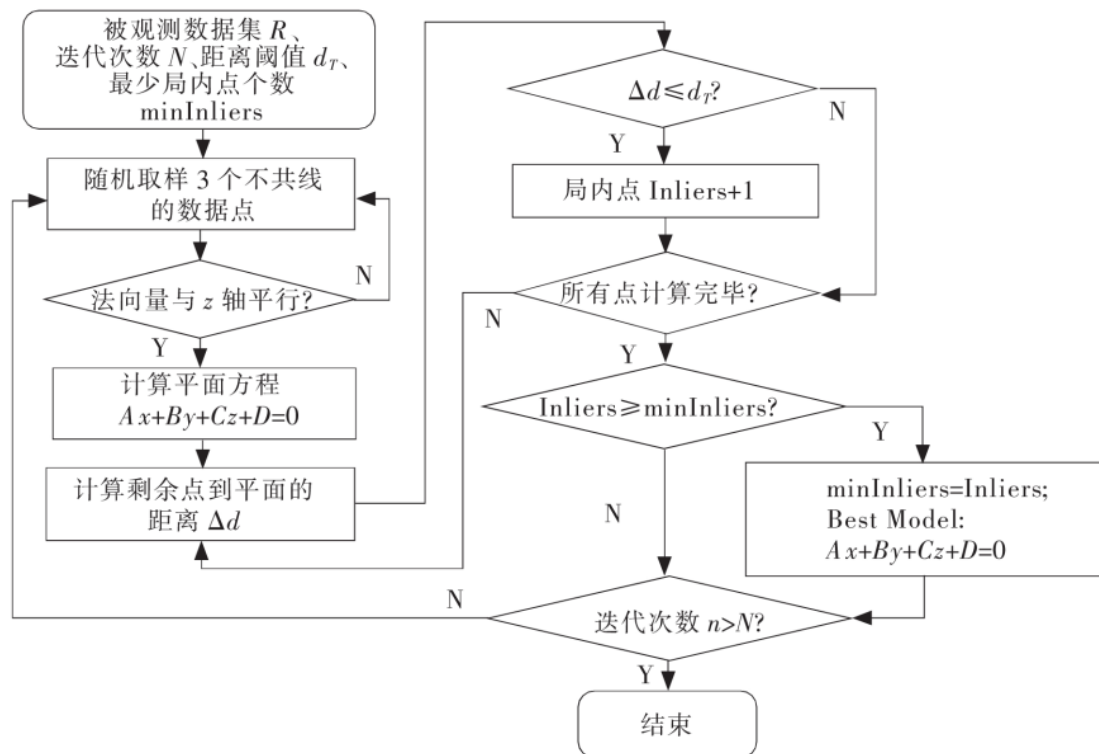
综上所述，在一幅点云  $C$  中提取线特征点和面特征点的流程为：

- (1) 遍历  $C$  的每一条扫描线，并计算每一个激光点的局部平滑度；
- (2) 对于  $C$  的第  $i$  根扫描线  $beam_i$ ，将  $beam_i$  等分为 6 段，并且在段内依据激光点的局部平滑度排序。每一段内选取局部平滑度大于线特征阈值的最大 2 个点作为线特征点，选

取局部平滑度小于面特征阈值的最小 4 个点作为面特征点，同时保证任意两个线特征点之间、任意两个面特征点之间的间隔均不能少于 5 个激光点；

(3) 对 C 所有扫描线均作步骤 (2) 的处理，最终获得点云的线特征点集合和面特征集合。

还有一些其他的点云分割和特征提取方法，基于 RANSAC 的地面点提取：



RANSAC 算法是一种鲁棒的模型拟合算法，其通过迭代方式，从一组包含离群点的被观测数据中估算出数学模型的参数。道路面是一个相对平坦的平面，通过 RANSAC 算法可以较好地实现路面模型提取。

PCL 点云库中也给出了一些更为复杂的点云特征描述子，比如点特征直方图描述子 (PFH)、快速点特征直方图描述子 (FPFH)、视点特征直方图描述子 (VFH)

[https://blog.csdn.net/zzh\\_AI/article/details/93589698](https://blog.csdn.net/zzh_AI/article/details/93589698)

## 2.2 前端扫描匹配

### 2.2.1 迭代最临近点及变种 ICP

参考文献：

1992\_Chen Yang\_Object modelling by registration of multiple range

2008\_Andrea Censi\_An ICP variant using a point-to-line metric

2004\_KL Low\_Linear Least-Squares Optimization for Point-to-Plane

由 Chen Yang 等人提出的 ICP 算法，利用待匹配的两帧点云欧式距离最小化，恢复相对位姿变换信息。ICP 方法分为已知对应点的求解和未知对应点的求解两种，其中已知对应点的情况能够直接计算出 R 和 T 的闭式解，而未知对应点的求解需要进行迭代计算，是 EM 算法的一个特例。Censi A 等人提出 ICP 变种算法（Point-to-Line ICP, PL-ICP），适用于 2D 激光 SLAM，PL-ICP 将当前帧数据根据初始位姿投影到参考坐标系，对当前帧的点在参考帧中找到最近的两个点，计算点到直线的距离误差，计算误差函数的迭代增量。Low K L 提出 ICP 变种算法（Point-to-Plane ICP，PP-ICP），适用于 3D 激光 SLAM。ICP、PL-ICP 与 PP-ICP 比较如表所示。

ICP 种类	误差函数	收敛速度	求解精度	不足
ICP	点对点距离	一阶收敛，较低	计算成本大	由于激光点的不连续性，ICP 会造成误匹配引入额外的误差，在退化环境中会迅速的累积误差，如长直的走廊。
Point-to-Line ICP	点到线距离	二阶收敛，较高	适用于 2D 激光 SLAM	PL-ICP 相比 ICP 对初始值更敏感，容易陷入局部极值；对于大旋转情况下算法不够鲁棒。
Point-to-Plane ICP	点到平面距离	二阶收敛，高	适用于 3D 激光 SLAM	需要从大量的三维激光点中提取特征点。



## 2.2.2 相关性扫描匹配 CSM

参考文献：

2009\_ Edwin B. Olson\_ Real-time correlative scan matching

2016\_ Wolfgang Hess\_ Real-Time Loop Closure in 2D LIDAR SLAM

由 Olson E B 等人提出的 CSM 算法，该算法匹配的似然场模型高度非凸，存在很多局部极值。由于进行暴力匹配，排除初值敏感的影响，这是与前面 ICP 方法的最大区别。CSM 算法流程如图所示，该方法是对环境分辨率细分的，精度会受限于分辨率。另外该方法能够通过加速策略来降低计算量，例如分枝定界方法。

谷歌开源方案 Cartographer 的前端采用的就是 CSM 与梯度优化结合，相关性扫描匹配的思路其实非常简单。想象给定一个**栅格地图**，再给定当前帧的点云，把激光雷达放在地图的每个格子上，看在这个位置时，点云是否与地图重合，重合程度最高的位置就是激光雷达的真实位姿。

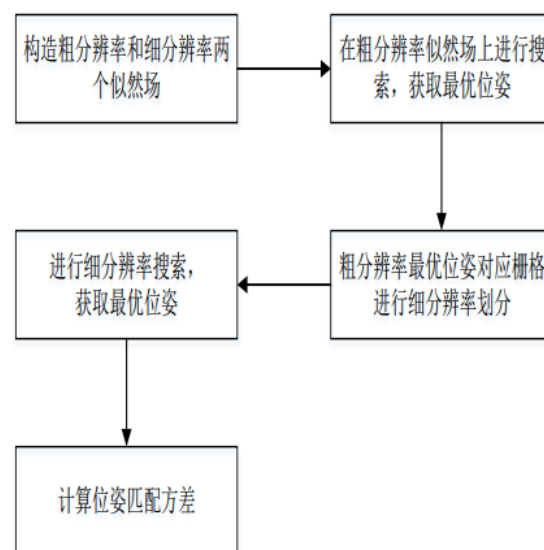


图 CSM 算法流程

[https://blog.csdn.net/changer\\_sun/article/details/79147446](https://blog.csdn.net/changer_sun/article/details/79147446)

占据栅格地图（Occupancy Grid Map）的概念：在通常的尺度地图中，对于一个点，它要么有（Occupied 状态）障碍物，要么没有（Free 状态），在占据栅格地图中，对于一个点，我们用  $p(s=1)$  来表示它是 Free 状态的概率，用  $p(s=0)$  来表示它是 Occupied 状态的概率，

两个值太多了，我们引入两者的比值来作为点的状态： $odd(s) = \frac{p(s=1)}{p(s=0)}$

对于一个点，新来了一个测量值（ $z \sim \{0,1\}$ ）之后我们需要更新它的状态。假设测量值

来之前，该点的状态为  $odd(s)$ ，我们要更新它为： $odd(s|z) = \frac{p(s=1|z)}{p(s=0|z)}$ 。这种表达方式

类似于条件概率，表示在  $z$  发生的条件下  $s$  的状态。

根据贝叶斯公式，我们有：

$$p(s=1|z) = \frac{p(z|s=1)p(s=1)}{p(z)}$$

$$p(s=0|z) = \frac{p(z|s=0)p(s=0)}{p(z)}$$

代入后，我们得到：

$$odd(s|z) = \frac{p(s=1|z)}{p(s=0|z)}$$

$$= \frac{p(z|s=1)p(s=1)}{p(z|s=0)p(s=0)}$$

$$= \frac{p(z|s=1)}{p(z|s=0)} odd(s)$$

这样，含有测量值的项就只剩下  $\frac{p(z|s=1)}{p(z|s=0)}$ ，我们称这个比值为测量值的模型，标记

为  $lomeas$ 。测量值的模型只含两种： $lofree = \log \frac{p(z=0|s=1)}{p(z=0|s=0)}$  和

$looccu = \log \frac{p(z=1|s=1)}{p(z=1|s=0)}$ ，而且都是定值。

这样，如果我们用  $\log(odd(s))$  来表示位置点的状态的话，我们的更新规则就进一步简

化成了： $S^+ = S^- + lomeas$ ，其中  $S^+$  和  $S^-$  分别表示测量之后和之前  $s$  的状态。另外，在没有任何测量值的初始状态下，一个点的初始状态：

$$\log(odd(s)) = \log \frac{p(s=1)}{p(s=0)} = \log \frac{0.5}{0.5} = 0$$

经过这样的建模，更新一个点的状态就只需要做简单的加减法了。例如，假设我们设定

$loccu = 0.9$  ,  $lofree = -0.7$  。那么， 一个点状态的数值越大，就表示越肯定它是 Occupied 状态，相反数值越小，就表示越肯定它是 Free 状态。

Cartographer 结合了 local 和 global 两种方式进行 2d SLAM。local 方式就是前端匹配中通过 submap 进行 scan matching。详见：

[https://blog.csdn.net/weixin\\_36976685/article/details/84994701?depth\\_1-utm\\_source=distribute.pc\\_relevant.none-task&utm\\_source=distribute.pc\\_relevant.none-task](https://blog.csdn.net/weixin_36976685/article/details/84994701?depth_1-utm_source=distribute.pc_relevant.none-task&utm_source=distribute.pc_relevant.none-task)

## A、Scans

一个scans即激光点云图，包含一个起点和许多的终点。起点称为origin，终点称为scan points，用  $H$  表示点云集，其表达形式如下。

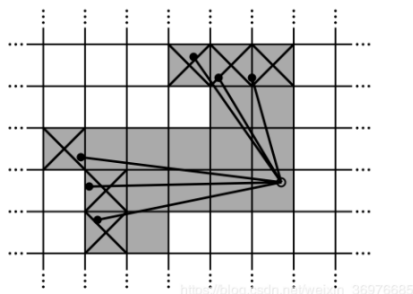
$$H = \{h_k\}_{k=1,\dots,K}$$

当获得一个新的scans，并且要插入到submap中时，scans中 $\{h_k\}$ 点集在submap中的位置被表示成 $T_\xi$ ，其转换公式如下：

$$T_\xi p = \underbrace{\begin{pmatrix} \cos \xi_\theta & -\sin \xi_\theta \\ \sin \xi_\theta & \cos \xi_\theta \end{pmatrix}}_{R_\xi} p + \underbrace{\begin{pmatrix} \xi_x \\ \xi_y \end{pmatrix}}_{t_\xi}.$$

## B、Submaps

一个submap是通过几个连续的scans创建而成的，由 $5cm \times 5cm$ 大小的概率栅格 $[p_{min}, p_{max}]$ 构造而成，submap在创建完成时，栅格概率小于 $p_{min}$ 表示该点无障碍，在 $p_{min}$ 与 $p_{max}$ 之间表示未知，大于 $p_{max}$ 表示该点有障碍。每一帧的scans都会生成一组称为hits的栅格点和一组称为misses的栅格点，如图所示。



## C、Ceres scan matching

每次获得的最新的scan需要插入到submap中最优的位置，使我们scan中的点束的位姿经过转换后落到submap中时，每个点的信度和最高。通过scan matching对 $T_\xi h_k$ 进行优化，这里的优化问题为解最小二乘问题，其问题描述可表示为：

$$\operatorname{argmin}_{\xi} \sum_{k=1}^K (1 - M_{\text{smooth}}(T_\xi h_k))^2 \quad (\text{CS})$$

其中 $M_{\text{smooth}}$ 是线性评价函数，方法为双三次插值法，该函数的输出结果为 $(0, 1)$ 以内的数，在这之外的数可以生成，但不被考虑进去，通过这种平滑函数的优化，能够提供比栅格分辨率更好的精度。该最小二乘问题在cartogrper中通过google自家的Ceres库进行求解。

## 2.2.3 正态分布变换 NDT

参考文献：

2003\_Peter Biber\_The Normal Distributions Transform A New Approach to Laser Scan Matching

2013\_Martin Magnusson\_The Three-Dimensional Normal-Distributions Transform (博士论文)

Biber P 等人提出的 NDT 算法，最开始作为一种二维的匹配方法，由 Magnusson M 等人将该算法应用到三维的匹配中，是把地图看成很多高斯分布的集合，不需要通过搜索，直接最小化目标函数便能得到转换关系，计算量小，速度较快。NDT 方法在 3D 激光 SLAM 与纯定位中使用较多。

为了知道当前载体在这个地图中的确切位置，我们比较激光雷达扫描得到的点云和地图点云，问题在于：激光雷达扫描得到的点云可能和地图的点云存在细微的区别，这里的偏差可能来自于测量误差，也有可能这个“场景”发生了一下变化（比如说行人，车辆）。NDT 配准就是用于解决这些细微的偏差问题。

NDT 并没有比较两个点云点与点之间的差距，而是先将参考点云（即高精度地图）转换为**多维变量的正态分布**，如果变换参数能使得两幅激光数据匹配的很好，那么变换点在参考系中的概率密度将会很大。因此，可以考虑用优化的方法求出使得概率密度之和最大的变换参数，此时两幅激光点云数据将匹配的很好。

NDT 算法流程：

**多元正态分布其概率密度函数可以表示为：**

$$f(\vec{x}) = \frac{1}{(2\pi)^{\frac{D}{2}} \sqrt{|\Sigma|}} e^{-\frac{(\vec{x}-\vec{\mu})^T \Sigma^{-1} (\vec{x}-\vec{\mu})}{2}}$$

D: 维度  $\vec{\mu}$  : 均值向量  $\Sigma$  : 协方差矩阵，我们知道，协方差矩阵对角元素表示的是对应的元素的方差，非对角元素则表示对应的两个元素（行与列）的相关性。

NDT 算法的第一步就是将参考点云网格化（对于三维地图来说，即使用一个一个的小立方体对整个空间的扫描点进行划分），对于每一个网格（cell），基于网格内的点计算其概率密度函数（probability density function, PDF），

$$\vec{\mu} = \frac{1}{m} \sum_{k=1}^m \vec{y}_k,$$
$$\Sigma = \frac{1}{m} \sum_{k=1}^m (\vec{y}_k - \vec{\mu})(\vec{y}_k - \vec{\mu})^T$$

其中， $\vec{y}_{k=1, \dots, m}$  表示一个网格内的所有扫描点。那么一个网格的概率密度函数则为：

$$f(\vec{x}) = \frac{1}{(2\pi)^{\frac{3}{2}} \sqrt{|\Sigma|}} e^{-\frac{(\vec{x}-\vec{\mu})^T \Sigma^{-1} (\vec{x}-\vec{\mu})}{2}}, \text{ 注意维数为 3, 表示 } x,y,z$$

使用正态分布来表示原本离散的点云有诸多好处，这种分块的（通过一个个 cell）光滑的表示是连续可导的，每一个概率密度函数可以被认为是一个局部表面（local surface）的近似，它不但描述了这个表面在空间中的位置，同时还包含了这个表面的方向和光滑性等信息，格子内少于 3 个点，经常会协方差矩阵不存在逆矩阵，所以只计算点数大于 5 的 cell，涉及到下采样方法。我们以三维的概率密度函数为例，如果三个特征值很接近，那么这个正态分布描述的表面是一个球面，如果一个特征值远大于另外两个特征值，则这个正态分布描述的是一条线，如果一个特征值远小于另外两个特征值，则这个正态分布描述的是一个平面。



当使用 NDT 配准时，目标是找到当前扫描的姿态，使当前扫描的点位于参考扫描（3D 地图）表面上的可能性最大化。那么我们需要优化的参数就是对当前扫描的点云的变换（旋转，平移等），我们使用一个变换参数  $\vec{p}$  来描述。当前扫描为一个点云  $X = \vec{x}_1, \dots, \vec{x}_n$ ，给定扫描点集合  $X$  和变换参数  $\vec{p}$ ，令空间转换函数  $T(\vec{p}, \vec{x}_k)$  表示使用使用姿态变换  $\vec{p}$  来移动点  $\vec{x}_k$ ，结合之前的一组状态密度函数（每个网格都有一个 PDF），那么最好的变换参数  $\vec{p}$  应该是最大化似然函数的姿态变换：

$$Likelihood : \Theta = \prod_{k=1}^n f(T(\vec{p}, \vec{x}_k))$$

最大化似然也就相当于最小化负对数似然， $-\log \Theta$

NDT 缺点：格子参数最重要，太大导致精度不高，太小导致内存过高，并且只有两幅图像相差不大的情况才能匹配

改进：八叉树建立，格子有大有小，迭代，每次使用更精细的格子；

K 聚类，有多少个类就有多少个 cell，格子大小不一；

三线插值 平滑相邻的格子 cell 导致的不连续，提高精度。

目前，日本的开源代码 Autoware 和国内的 Apollo 都采用了基于 NDT 的定位模块。由于 NDT 算法不需要匹配各个点计算速度较 ICP 快，Autoware 官方建议定位模块使用 NDT 算法，然后通过使用 CUDA 实现的 fast\_pcl package 实现了对 NDT 优化过程的并行加速。

[https://blog.csdn.net/orange\\_littlegirl/article/details/89262501?depth\\_1-](https://blog.csdn.net/orange_littlegirl/article/details/89262501?depth_1-)

[utm\\_source=distribute.pc\\_relevant.none-task&utm\\_source=distribute.pc\\_relevant.none-task](utm_source=distribute.pc_relevant.none-task&utm_source=distribute.pc_relevant.none-task)

## 2.2.4 其他

参考文献：

2018\_J E Deschaud\_IMLS-SLAM scan-to-model matching based on 3D data

2016\_Austin Nicolai\_Deep Learning for Laser Based Odometry Estimation

2019\_Cyrill Stachniss\_SuMa++ Efficient LiDAR-based Semantic SLAM

Deschaud J E 等人基于特定的采样策略和扫描到模型（scan-to-model）的匹配方式提出一种纯 3D 激光 SLAM 算法。

在结合深度学习方面，Nicolai A 等人利用 VLP-16 采集 3D 点云数据，将其投影到 2D 平面生成深度图像，利用 CNN 网络训练，得到端到端的匹配结果，其运行速度明显快于传统 ICP 匹配方法。

Cyrill Stachniss 等人基于深度学习的卷积神经网络，通过语义分割激光雷达点云来获取点云级的密集语义信息，基于该语义约束来进一步提高投影匹配 ICP 的位姿估计精度。

## 2.3 后端优化

### 2.3.1 基于滤波器

基于滤波器的激光 SLAM 是一个贝叶斯估计的过程。下面对几种滤波的形式进行总结

滤波器形式	描述
卡尔曼滤波 (KF)	递归的线性高斯系统最优估计。
扩展卡尔曼滤波 (EKF)	当非线性非高斯系统时, 将在工作点附近近似为线性高斯进行处理。
迭代卡尔曼滤波 (IEKF)	对工作点进行迭代。
无迹卡尔曼滤波 (UKF)	没有线性化近似, 而是把 $\sigma$ 点进行非线性变换后再用高斯分布近似。
粒子滤波 (PF)	去掉高斯假设, 直接采用蒙特卡洛的方式, 以粒子作为采样点来描述分布 <sup>[24]</sup> 。

在基于滤波器的激光 SLAM 中常常采用粒子滤波 (PF) 作为数学优化的框架。用粒子滤波来估计机器人位姿, 将每个粒子用运动学模型进行传播, 对于传播后的粒子用观测模型进行权重计算并根据估计的位姿构建地图。该方案存在两个问题, 第一, 由于每个粒子包含机器人的轨迹和对应的环境地图, 对于大尺度环境, 若里程计误差较大即预测分布与真实分布差异较大, 则需要较多粒子来表示机器人位姿的后验概率分布, 严重消耗内存; 第二, 由于重采样的随机性, 随着重采样次数增多, 粒子多样性散失, 粒子耗散问题会严重影响地图的构建。早期的激光 SLAM 多采用粒子滤波进行优化, 如 2002 年 FastSLAM, 2007 年 Gmapping, 2010 年 Optimal RBPF。

### 2.3.2 基于优化

利用图优化 (graph-based optimization) 的数学框架优化 SLAM 问题, 通过非线性最小二乘方法来优化建图过程中累积的误差。SLAM 框架目前多采用基于优化的方法。

Konolige K 等人提出首个基于图优化框架的开源方案 Karto SLAM, 该方案认识到了系统稀疏性, 在一定程度上替代了基于滤波器的激光 SLAM 方案。该方案的不足是采用局部子图匹配之前都要构建子图, 耗费时间较长; 若采用全局匹配方法, 则在搜索范围大的时候速度会变慢。谷歌的 Cartographer 开源方案, 是对 Karto SLAM 的优化方案, 核心内容是融合多传感器数据的局部子图创建以及用于闭环检测的扫描匹配策略。该方案中前端扫描匹配算法是结合 CSM 与梯度优化来实现的。在生成一个子地图后, 会进行一次局部的闭环检测; 当全部子地图构建完成后, 利用分枝定界和预先计算的网格的算法, 进行全局闭环检测, 从而保证闭环检测的速度。该方案的不足是没有对闭环检测结果进行验证, 在几何对称的环境中, 容易引起错误的闭环。

## 2.4 回环检测

随着无人车行驶距离的增加，经过配准的位姿仍然会逐渐产生漂移。在视觉 SLAM 中，减小大范围、长距离定位误差的有效方法是进行闭环检测和图优化。对于闭环检测，视觉 SLAM 中普遍采用 BoW 算法，而针对激光雷达点云尚无有效的闭环检测方法。

### 2.4.1 基于几何距离

参考：LeGO-LAOM

2018\_Tixiao Shan\_LeGO-LOAM Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain

一个可以考虑的方案是基于位置的闭环检测，即将历史位置保存在 3D K-D 树中，然后每增加一个位姿（位置和姿态）都在 K-D 树中搜索距离当前位置最近的历史位置，如果两个位置距离小于一定阈值则判定形成闭环。但该方案可行的前提是历史位置相对于真值的偏差不能过大，否则无法成功判断闭环或者产生错误的闭环。

### 2.4.2 基于点云形状

参考：segMatch

2019\_Renaud Dubé\_SegMatch Segment Based Place Recognition in 3D Point Clouds

segmatch 是一个提供车辆的回环检测的技术，使用提取和匹配分割的三维激光点云技术。分割的例子可以在下面的图片中看到。



该技术是基于在车辆附近提取片段（例如车辆、树木和建筑物的部分），并将这些片段与从目标地图中提取的片段相匹配。分段匹配可以直接转化为精确的定位信息，从而实现精确的三维地图构造和定位。该方法依赖于分割对象但不一定限于语义对象，因为这允许更一般的表示分割物，并在更广泛的不同环境中启用功能。

对于理想情况下的回环一般满足两个假设：第一是能够应用分割技术将物体分割出来，第二是环境中必须有物体对象。它首先第一步是从获得的 3D 点云中提取并描述分割物，将



它们匹配用于已经走过的地方和使用几何验证对比的方法选出回环检测的候选点云。这种基于分割的技术的优点是将点云压缩成一组清晰的用于闭环检测的判别元素。实验表明这不仅减少了匹配所需的时间，也减少了错误匹配的可能性。

## 2.5 重点与难点

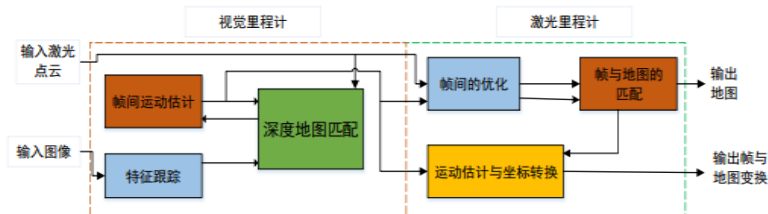
1. 点云分割和目标检测：给点云聚类标注，是点云领域的热点问题；用于目标检测、动态物体滤除、基于点云分割的匹配技术，是深度学习的基础；还有很多文章讨论了地面、车道线、路沿提取等。深度学习在 3D 点云场景中的应用：是针对目标检测、场景分割等任务，PointNet 系列基本上是近两年来点云分割网络的 baseline，其优点是参数量较小，缺点是对局部的特征抓取还不是很完善。
2. 回环检测：LeGO-LOAM 是基于几何位置的回环检测，缺点有，一是相隔稍远的关键帧之间点云重合度太小从而无法建立边的约束，二是 ICP 配准时陷入局部最优的情况，换成 NDT 会不会好一点？Segmatch 这种基于点云形状匹配的回环检测是一种好方法，loam\_livox 基于帧间二维直方图的归一化互相关进行相似度评价听起来也不错。
3. SLAM 算法的鲁棒性与实时性：在提高 SLAM 算法的鲁棒性方面，需要考虑里程计的标定、激光雷达的外参与时间戳标定、激光雷达运动畸变的去除等数据处理过程，同时针对退化环境（环境空旷，无法定位）、全局定位、动态环境定位等问题还有待完善。
4. 与其他传感器数据融合，比如相机、GPS 接收机；视觉会提供高精度的里程计以及信息量丰富的地图信息，激光雷达为视觉特征提供准确的深度信息；卫星导航能提供准确的全局坐标，激光雷达在卫星导航失锁时进行航位推算。
5. 其他重点发展方向：多机器人协作的场景，同时定位与建图；自动驾驶地图，地图的动态更新；智能车辆的障碍物检测与跟踪；

# 3LeGO-LOAM 开源代码详细讲解

## 3.0 基于 LOAM 开发的代码

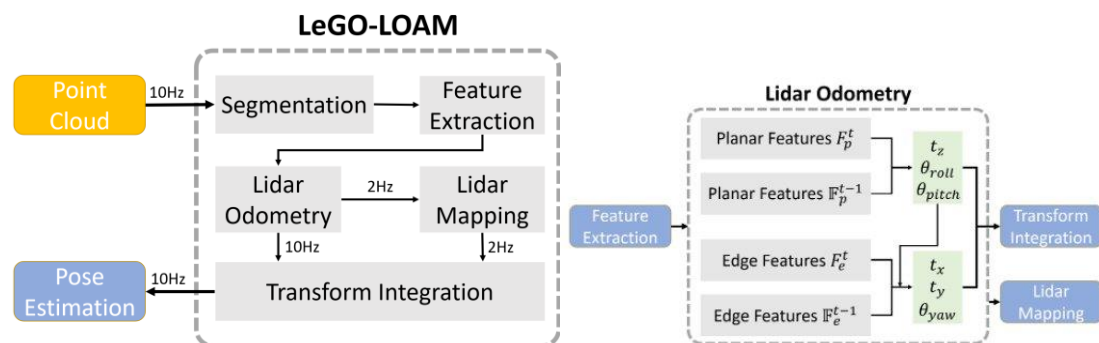
原作者论文：2017\_Ji Zhang\_Low-drift and Real-time Lidar Odometry and Mapping

名称	改进
----	----

<b>LeGO-LOAM</b>	引入了回环检测模块，采用两步优化的方式，降低设计矩阵的维数，可以满足实时性的要求
<a href="#"><u>A-LOAM</u></a>	使用 Eigen 和 Ceres Solver 库简化了原 LOAM 代码
<a href="#"><u>loam_livox</u></a>	专为无人机上的小视场角激光雷达开发的包，回环检测计算当前关键帧的二维直方图，并使用二维直方图的归一化互相关作为与地图中关键帧之间的相似性度量。
<a href="#"><u>LOAM-multi-thread</u></a>	将原来的 LOAM 调整为不需要 ROS 的多线程版本，可以在 MacOS、Windows 等 ROS 不方便的多平台下运行。
<a href="#"><u>LLS-LOAM</u></a>	Lidar Odometry and Mapping with Mutiple Metrics Linear Least Square ICP，使用线性最小二乘法进行 ICP 配准
<b>V-LOAM（原作者的改进）</b>	

### 3.1 代码结构

代码是基于 ROS 系统写的，主要有 4 个.cpp 文件（即 4 个节点），1 个.h 文件，一个 launch 节点文件（同时开启四个节点），一个点云 message 定义文件。Message 是节点间互相传递的数据，在 ROS 中我们得给采集到的消息取个名字用来区分不同的 message，这就是 topic 了，topic 会用/xxxxxx 表示。



#### Utility.h

定义了一些公用的东西，激光雷达参数、函数参数、变量阈值等

#### ImageProjection.cpp

主要是为了对三维点云进行投影，以及点云分割。

最重要的函数，有关激光雷达点云数据的回调函数：

```
void cloudHandler(const sensor_msgs::PointCloud2ConstPtr& laserCloudMsg){  
    copyPointCloud(laserCloudMsg); // 将 rosmmsg 转化为 pcl 点云  
    findStartEndAngle(); // 寻找始末角度  
    projectPointCloud(); // 点云投影，构建一个距离矩阵 rangeMat  
    groundRemoval(); // 可能表示地面的扫描点被标记为地面点  
    cloudSegmentation(); // 点云聚类分簇  
    publishCloud();  
    resetParameters();  
}
```

### **featureAssociation.cpp**

包括特征提取匹配和位姿估计两大部分

FeatureAssociation 的构造函数初始化了一些 ros 话题的收发，接着就是一个死循环不断的调用 runFeatureAssociation。转到构造函数，发现它接收上一个节点的分割点云、异常值以及 IMU 信息，也发送了很多点云的话题，供建图、可视化用。

runFeatureAssociation 函数：

```
adjustDistortion();// 进行坐标系的调整，velodyne lidar 被安装为 x 轴向前,y 轴向左,z 轴  
向上的右手坐标系，通过交换坐标轴，都统一到 z 轴向前,x 轴向左,y 轴向上的载体坐标系  
calculateSmoothness(); // 检测了点云特征点的曲率  
markOccludedPoints(); // 去除一些瑕点  
extractFeatures(); // 提取特征  
publishCloud(); // 发布特征点云用于可视化  
// 位姿估计  
if (!systemInitLM) {  
    checkSystemInitialization();// 将当前时刻保存的 IMU 数据作为先验数据  
    return;  
}  
updateInitialGuess();// 提供粗配准的先验以供优化估计
```

```
updateTransformation();// 分别对平面特征与角点特征进行了匹配计算，首先找到各自对应的特征，平面特征与角部特征的计算类似，都使用了 LM 法进行迭代  
integrateTransformation();// 将帧间运动量转换到世界坐标系下  
publishOdometry();  
publishCloudsLast(); // 发送点云以供建图使用
```

### MapOptimization.cpp

mapOptimization.cpp 进行的内容主要是地图优化，将得到的局部地图信息融合到全局地图中去。

main()函数的关键代码就三条，也就是三个不同的线程，最重要的是 run()函数：

**mapOptimization MO;** //构造 mapOptimization 对象 MO, 构造函数定义了该节点要订阅和发布的主题

**std::thread loopthread(&mapOptimization::loopClosureThread, &MO);**

// 进行闭环检测与闭环的功能，将 MO 作为参数传入构造的线程中使用

**std::thread visualizeMapThread(&mapOptimization::visualizeGlobalMapThread, &MO);**

// 该线程中进行的工作是 publishGlobalMap(),将数据发布到 ros 中，可视化

**MO.run();** //主函数，下面详细介绍

transformAssociateToMap();// 把点云坐标均转换到世界坐标系下

extractSurroundingKeyFrames();// 由于帧数的频率大于建图的频率，因此需要提取关键帧进行匹配，抽取当前 keyframe 附近的关键帧构造点云地图

downsampleCurrentScan();// 降采样当前帧点云

scan2MapOptimization();// 当前关键帧和点云地图进行配准，优化位姿估计

saveKeyFramesAndFactor();// 保存当前关键帧的位姿

//发送 TF 变换可视化

correctPoses();

publishTF();

publishKeyPosesAndFrames();

clearCloud();

以上是主线程的大致流程，接下来看看其他线程的工作：

一、 `visualizeGlobalMapThread` 是以 0.2Hz 的频率发布地图点云，它对地图点云进行了两次降维，并且将角点、平面点、异常点均加入到最终的地图点云中发布。

主要进行了 3 个步骤：

- 1、通过 `KDTree` 进行最近邻搜索；对当前位置范围 500m 内进行搜索
- 2、通过搜索到关键帧的索引放进队列；
- 3、第一次降采样关键帧轨迹，第二次降采样点云集合，通过两次下采样，减小数据量；

二、 `loopClosureThread` 是 1HZ 的频率在回环检测。`LeGO-LOAM` 的回环检测是基于位置也就是里程计的几何关系来实现的。

`performLoopClosure()`，采用经典 ICP 点云配准，最终匹配得到坐标转换。在位姿图中加入新匹配得到的位姿约束，用 `GTSAM` 进行优化，回环检测成功后，位姿图会在 `saveKeyFramesAndFactor` 函数中依次更新，这样就实现了回环检测的校正功能，而地图点云也会再次更新，这样我们就实现了建图功能。

## TransformFusion.cpp

最后这个节点负责可视化位姿，融合里程计输出和地图优化后输出的位姿。主要是两个回调函数，`laserOdometryHandler` 和 `odomAftMappedHandler`。一旦接收到里程计的输出，就立即把轨迹点显示在屏幕上，这个是实时的；另外接收到地图优化的输出，重新调整轨迹的显示，这个是滞后的。

## 3.2 基础理论

### 3.2.1 点云分割与特征提取

同 [2.1.2](#)

### 3.2.2 激光雷达里程计

激光 SLAM 系统通过提取激光点云中的线特征和面特征，匹配前后两帧点云的线面特征，从而恢复运动估计进行航位推算。考虑到提取特征后的激光点云是非常稀疏的，当前帧

的某特征点不一定能找到参考帧里的同名点，即利用 K-D 树搜索出来的最近邻点可能并不指代同一个点，坐标观测值存在误差。因为线特征点只和线特征点进行匹配，面特征点只和面特征点进行匹配。对于线特征点而言，虽然最近邻点和它可能不是同一个点，但是它们在同一条线上的可能性很大，所以观测方程变成了最小化点到线/面的距离之和的函数。

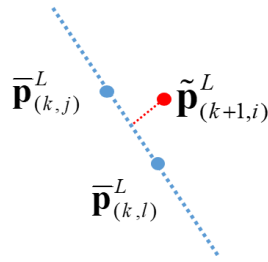
设第  $k$  帧的原始点云  $C_k$  已经过去畸变处理投影到  $\{L_{k+1}\}$  下得到  $\bar{C}_k$ ，去畸变后的线特征点集合为  $\bar{C}_{line,k}$ ，面特征点集合为  $\bar{C}_{plane,k}$ ，设已经求得第  $k$  帧点云内的车体运动量为  $TR_k$ 。此时给定第  $k+1$  帧原始点云  $C_{k+1}$ ，其线特征集合为  $C_{line,k+1}$ ，面特征集合为  $C_{plane,k+1}$ ，帧头扫描时刻为  $t_{k+1}$ ，帧尾扫描时刻为  $t_{k+2}$ 。我们通过如下的步骤求解时刻  $t_{k+1}$  到  $t_{k+2}$  时刻内的运动量  $TR_{k+1}$

(1) 用  $TR_k$  初始化  $TR_{k+1}$ ，是因为基于车体匀速运动的假设；

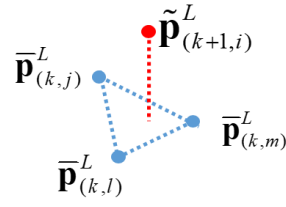
(2) 根据位姿插值的方法将  $C_{line,k+1}$ ， $C_{plane,k+1}$  投影到  $\{L_{k+1}\}$  下，得到  $\tilde{C}_{line,k+1}$ ， $\tilde{C}_{plane,k+1}$ ；

$$\tilde{\mathbf{p}}_{(k+1,i)}^L = S \cdot TR_{k+1} \cdot \mathbf{p}_{(k+1,i)}^L = \mathbf{R}_{(k+1,i)}^{-1} (\mathbf{p}_{(k+1,i)}^L - \mathbf{T}_{(k+1,i)})$$

(3) 特征点最近邻匹配，计算误差函数。对于  $k+1$  帧的每一个线特征点，在第  $k$  帧中找到两个与之匹配的线特征点（两点构成一条直线），对于  $k+1$  帧的每一个面特征点，在第  $k$  帧中找到三个与之匹配的面特征点（三点构成一个平面）



线特征点匹配



面特征点匹配

基于 kd 树的搜索策略，对于  $\tilde{C}_{line,k+1}$  中的每一个线特征点  $\tilde{\mathbf{p}}_{(k+1,i)}^L$ ，在  $\bar{C}_{line,k}$  中查找两个距离  $\tilde{\mathbf{p}}_{(k+1,i)}^L$  最近的匹配点，记作  $\bar{\mathbf{p}}_{(k,j)}^L$  和  $\bar{\mathbf{p}}_{(k,l)}^L$ ，并且由于一条扫描线在一条脊线上只能得到一个线特征点，因此还需要保证  $\bar{\mathbf{p}}_{(k,j)}^L$  和  $\bar{\mathbf{p}}_{(k,l)}^L$  不属于同一条扫描线。得到两个配对点之后，可以计算  $\tilde{\mathbf{p}}_{(k+1,i)}^L$  距离  $\bar{\mathbf{p}}_{(k,j)}^L$  和  $\bar{\mathbf{p}}_{(k,l)}^L$  构成直线的距离，如式所示：

$$d = \frac{|(\tilde{\mathbf{p}}_{(k+1,i)}^L - \bar{\mathbf{p}}_{(k,j)}^L) \times (\tilde{\mathbf{p}}_{(k+1,i)}^L - \bar{\mathbf{p}}_{(k,l)}^L)|}{|(\bar{\mathbf{p}}_{(k,j)}^L - \bar{\mathbf{p}}_{(k,l)}^L)|}$$

同理，对于  $\tilde{C}_{plane,k+1}$  中的每一个面特征点  $\tilde{\mathbf{p}}_{(k+1,i)}^L$ ，在  $\bar{C}_{line,k}$  中查找三个距离  $\tilde{\mathbf{p}}_{(k+1,i)}^L$  最近的匹配点，记作  $\bar{\mathbf{p}}_{(k,j)}^L$ ， $\bar{\mathbf{p}}_{(k,l)}^L$ ， $\bar{\mathbf{p}}_{(k,m)}^L$ ，并且共线的三个点无法组成平面，因此还需要保证  $\bar{\mathbf{p}}_{(k,j)}^L$ ， $\bar{\mathbf{p}}_{(k,l)}^L$ ， $\bar{\mathbf{p}}_{(k,m)}^L$ ，不属于同一条扫描线。得到三个配对点之后，可以计算  $\tilde{\mathbf{p}}_{(k+1,i)}^L$  距离  $\bar{\mathbf{p}}_{(k,j)}^L$ ， $\bar{\mathbf{p}}_{(k,l)}^L$ ， $\bar{\mathbf{p}}_{(k,m)}^L$  构成的面的距离，如式所示：

$$d = \frac{|(\tilde{\mathbf{p}}_{(k+1,i)}^L - \bar{\mathbf{p}}_{(k,j)}^L) \cdot ((\bar{\mathbf{p}}_{(k,j)}^L - \bar{\mathbf{p}}_{(k,l)}^L) \times (\bar{\mathbf{p}}_{(k,j)}^L - \bar{\mathbf{p}}_{(k,m)}^L))|}{|(\bar{\mathbf{p}}_{(k,j)}^L - \bar{\mathbf{p}}_{(k,l)}^L) \times (\bar{\mathbf{p}}_{(k,j)}^L - \bar{\mathbf{p}}_{(k,m)}^L)|}$$

由上式可得线、面特征点与匹配点的误差函数：

$$d = f(\mathbf{TR}_{k+1})$$

(4) 使用算法更新  $\mathbf{TR}_{k+1} = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z]^T$ 。计算  $f$  相对  $\mathbf{TR}_{k+1}$  的雅克比矩阵  $\mathbf{J}$ ：

$$\mathbf{J} = \partial f / \partial \mathbf{TR}_{k+1}$$

则可以对  $\mathbf{TR}_{k+1}$  做如下更新：

$$\mathbf{TR}_{k+1} \leftarrow \mathbf{TR}_{k+1} - \left( \mathbf{J}^T \mathbf{J} + \lambda \mathbf{diag}(\mathbf{J}^T \mathbf{J}) \right)^{-1} \mathbf{J}^T d$$

其中  $\lambda$  是由 Levenberg-Marquardt 算法确定的系数。

补充：

LOAM 中定义的载体坐标系是 X 向左、Y 向上、Z 向前，旋转矩阵的旋转顺序是 **YXZ**，**转角顺时针为正**， $\mathbf{TR}_{k+1} = \mathbf{R}_z(\theta_z) \mathbf{R}_x(\theta_x) \mathbf{R}_y(\theta_y)$ ， $\mathbf{TR}_{k+1}^{-1} = \mathbf{R}_y(-\theta_y) \mathbf{R}_x(-\theta_x) \mathbf{R}_z(-\theta_z)$ 。

接下来推导雅各比矩阵，使用链式法则：

$$J_i = \frac{\partial f}{\partial \mathbf{TR}_{k+1}} = \frac{\partial f}{\partial \tilde{\mathbf{p}}^T} \frac{\partial \tilde{\mathbf{p}}}{\partial \mathbf{TR}_{k+1}}$$

$$1) \text{ 求左半部分: } \frac{\partial f}{\partial \tilde{\mathbf{p}}^T}$$

$$\frac{\partial f}{d\tilde{p}^T} = \begin{bmatrix} \frac{\partial f}{dx_0} & \frac{\partial f}{dy_0} & \frac{\partial f}{dz_0} \end{bmatrix}$$

原始点记为  $p$ ，变换到扫描开始时刻为  $\tilde{p}$ ，最近线段的 2 个端点为  $p_1, p_2$ ，

$$d = \frac{|\tilde{p}p_1 \times \tilde{p}p_2|}{|p_1p_2|} \quad p_1 = \begin{bmatrix} x_1 & y_1 & z_1 \end{bmatrix} \quad p_2 = \begin{bmatrix} x_2 & y_2 & z_2 \end{bmatrix} \quad \tilde{p} = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}$$

$$\tilde{p}p_1 \times \tilde{p}p_2 = \begin{bmatrix} m_{11} \\ m_{22} \\ m_{33} \end{bmatrix} = \begin{bmatrix} (y_0 - y_1)*(z_0 - z_2) - (y_0 - y_2)*(z_0 - z_1) \\ (x_0 - x_2)*(z_0 - z_1) - (x_0 - x_1)*(z_0 - z_2) \\ (x_0 - x_1)*(y_0 - y_2) - (x_0 - x_2)*(y_0 - y_1) \end{bmatrix}$$

设：

$$a_{012} = \sqrt{m_{11} * m_{11} + m_{22} * m_{22} + m_{33} * m_{33}}$$

$$l_{12} = \sqrt{p_1p_2} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

则：(1a 表示 f 对 x0 求偏导，1b 表示 f 对 y0 求偏导，1c 表示 f 对 z0 求偏导)

$$1a = \frac{\partial f}{dx_0} = \frac{m_{33} * (y_1 - y_2) - m_{22} * (z_1 - z_2)}{a_{012} * l_{12}}$$

$$1b = \frac{\partial f}{dy_0} = \frac{-m_{33} * (x_1 - x_2) - m_{11} * (z_1 - z_2)}{a_{012} * l_{12}}$$

$$1c = \frac{\partial f}{dz_0} = \frac{m_{22} * (x_1 - x_2) + m_{11} * (y_1 - y_2)}{a_{012} * l_{12}}$$

同理，原始点记为  $p$ ，变换到扫描开始时刻为  $\tilde{p}$ ，最近平面的 3 个端点为  $p_1, p_2, p_3$

$$d = \frac{|\tilde{p}p_1 \cdot (p_1p_2 \times p_1p_3)|}{|p_1p_2 \times p_1p_3|}$$

$$p_1 = \begin{bmatrix} x_1 & y_1 & z_1 \end{bmatrix} \quad p_2 = \begin{bmatrix} x_2 & y_2 & z_2 \end{bmatrix} \quad p_3 = \begin{bmatrix} x_3 & y_3 & z_3 \end{bmatrix} \quad \tilde{p} = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}$$

记：

$$|p_1p_2 \times p_1p_3| = \begin{bmatrix} p_a \\ p_b \\ p_c \end{bmatrix} = \begin{bmatrix} (y_1 - y_2)*(z_1 - z_3) - (y_1 - y_3)*(z_1 - z_2) \\ (x_1 - x_3)*(z_1 - z_2) - (x_1 - x_2)*(z_1 - z_3) \\ (x_1 - x_2)*(y_1 - y_3) - (x_1 - x_3)*(y_1 - y_2) \end{bmatrix}$$

$$\tilde{p}p_1 = \begin{bmatrix} x_1 - x_0 & y_1 - y_0 & z_1 - z_0 \end{bmatrix}$$

那么：



$$d = \frac{(z_0 - z_1) * p_c + (y_0 - y_1) * p_b + (x_0 - x_1) * p_a}{\sqrt{p_a^2 + p_b^2 + p_c^2}}$$

$$\frac{\partial f}{\partial x_0} = \frac{p_a}{\sqrt{p_a^2 + p_b^2 + p_c^2}}$$

$$\frac{\partial f}{\partial y_0} = \frac{p_b}{\sqrt{p_a^2 + p_b^2 + p_c^2}}$$

$$\frac{\partial f}{\partial z_0} = \frac{p_c}{\sqrt{p_a^2 + p_b^2 + p_c^2}}$$

$$2) \text{ 求右半部分 } \frac{\partial \tilde{p}}{\partial \mathbf{TR}_{k+1}}$$

$$\frac{\partial \tilde{p}}{\partial \mathbf{TR}_{k+1}} = \begin{bmatrix} \frac{\partial x_0}{\partial \theta_x} & \frac{\partial x_0}{\partial \theta_y} & \frac{\partial x_0}{\partial \theta_z} & \frac{\partial x_0}{\partial t_x} & \frac{\partial x_0}{\partial t_y} & \frac{\partial x_0}{\partial t_z} \\ \frac{\partial y_0}{\partial \theta_x} & \frac{\partial y_0}{\partial \theta_y} & \frac{\partial y_0}{\partial \theta_z} & \frac{\partial y_0}{\partial t_x} & \frac{\partial y_0}{\partial t_y} & \frac{\partial y_0}{\partial t_z} \\ \frac{\partial z_0}{\partial \theta_x} & \frac{\partial z_0}{\partial \theta_y} & \frac{\partial z_0}{\partial \theta_z} & \frac{\partial z_0}{\partial t_x} & \frac{\partial z_0}{\partial t_y} & \frac{\partial z_0}{\partial t_z} \end{bmatrix}$$

又因为，

$$\tilde{p} = \mathbf{R}_{(k+1,i)}^{-1} (\mathbf{p} - \mathbf{T}_{(k+1,i)})$$

记  $r_x = s * \theta_x$     $r_y = s * \theta_y$     $r_z = s * \theta_z$  ,  $s$  表示某点在一帧扫描内的相对时间，详见点

云运动畸变去除的公式，对每个点做姿态内插：

$$\tilde{p} = \begin{pmatrix} \cos(r_y) & 0 & -\sin(r_y) \\ 0 & 1 & 0 \\ \sin(r_y) & 0 & \cos(r_y) \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(r_x) & \sin(r_x) \\ 0 & -\sin(r_x) & \cos(r_x) \end{pmatrix} \begin{pmatrix} \cos(r_z) & \sin(r_z) & 0 \\ -\sin(r_z) & \cos(r_z) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x - st_x \\ y - st_y \\ z - st_z \end{pmatrix}$$

求导结果和 LOAM 的推导一致。

3) 综上，

$$J_i = \frac{\partial f}{\partial \tilde{p}^T} \frac{\partial \tilde{p}}{\partial \mathbf{TR}_{k+1}} = [l_a \quad l_b \quad l_c] \begin{bmatrix} \frac{\partial x_0}{\partial \theta_x} & \frac{\partial x_0}{\partial \theta_y} & \frac{\partial x_0}{\partial \theta_z} & \frac{\partial x_0}{\partial t_x} & \frac{\partial x_0}{\partial t_y} & \frac{\partial x_0}{\partial t_z} \\ \frac{\partial y_0}{\partial \theta_x} & \frac{\partial y_0}{\partial \theta_y} & \frac{\partial y_0}{\partial \theta_z} & \frac{\partial y_0}{\partial t_x} & \frac{\partial y_0}{\partial t_y} & \frac{\partial y_0}{\partial t_z} \\ \frac{\partial z_0}{\partial \theta_x} & \frac{\partial z_0}{\partial \theta_y} & \frac{\partial z_0}{\partial \theta_z} & \frac{\partial z_0}{\partial t_x} & \frac{\partial z_0}{\partial t_y} & \frac{\partial z_0}{\partial t_z} \end{bmatrix}$$

(5) 重复步骤(2)~(4)进行迭代，直到两次  $\mathbf{TR}_{k+1}$  的变化值足够小则结束迭代。

经过上述 5 个步骤，我们已经计算出了第  $k+1$  帧内的车体运动量  $\mathbf{TR}_{k+1}$ ，此时用  $\mathbf{TR}_{k+1}$

将第  $k+1$  帧原始点云  $C_{k+1}$  经过去畸变处理投影在  $\{L_{k+2}\}$ ，为下一帧特征点配准作为参考。

### 3.2.3 地图优化

我们对于  $\mathbf{TR}_k$  的估计过程中仅仅考虑了第  $k$  帧和第  $k-1$  帧点云的信息，因此估计结果必然存在误差，如果不加以修正则上述累积位姿的方法将使得帧间误差同样被累积，导致帧数较大时车体的位姿估计产生明显漂移。为减少误差的累积，我们不仅需要考虑帧间点云的配准，还需要将当前帧点云与历史点云地图进行配准。

地图优化的过程具体如下：

(1) 设当前时刻为  $t_{k+1}$ ，第  $k$  帧点云是关键帧，线特征点与面特征点投影在  $\{L_{k+1}\}$  坐标系下得到  $\bar{C}_k, \bar{C}_{line,k}, \bar{C}_{plane,k}$ ，当前的初始位姿估计为  $\mathbf{Pose}_k$ ，同时设此时地图中保存了第  $k$  帧之前的所有关键帧的点云  $\mathbf{Map}$

(2) 特征点投影至世界坐标系，将  $\bar{C}_{line,k}$  和  $\bar{C}_{plane,k}$  投影到世界坐标系  $\{W\}$  下，记作  $C_{line,k}^W$ ， $C_{plane,k}^W$

(3) 最近邻匹配，计算误差函数。遍历  $C_{line,k}^W$  和  $C_{plane,k}^W$  中的每一个特征点，对每一个特征点分别在  $\mathbf{Map}$  中找到匹配的直线和平面，并计算点到线距离或点到面距离。

这里的最近邻搜索和里程计部分的不同，是查找最近邻的 5 个点，采用主成分分析法验

证这 5 个点是否呈直线分布，或者 QR 求解法向量验证这 5 个点是否呈平面分布。

(4) 使用算法更新  $\mathbf{Pose}_k$ 。计算  $f$  相对  $\mathbf{Pose}_k$  的雅克比矩阵  $\mathbf{J}$ ：

$$\mathbf{J} = \partial f / \partial \mathbf{Pose}_k$$

则可以对  $\mathbf{Pose}_k$  做如下更新：

$$\mathbf{Pose}_k \leftarrow \mathbf{Pose}_k - \left( \mathbf{J}^T \mathbf{J} + \lambda \mathbf{diag}(\mathbf{J}^T \mathbf{J}) \right)^{-1} \mathbf{J}^T d$$

其中  $\lambda$  是由 LM 算法确定的系数。

(5) 重复步骤(2)~(4)进行迭代，直到  $\mathbf{Pose}_k$  两次的变化值足够小则结束迭代。

经过步骤 (1) ~ (5) 我们得到了与历史地图配准过的位姿估计，记作  $\mathbf{Pose}'_k$ 。相比于初始位姿估计值， $\mathbf{Pose}'_k$  不仅考虑了与相邻帧点云配准，还与历史地图进行了配准，因而减小了累积误差。获得更为准确地  $\mathbf{Pose}'_k$  之后，我们将  $\bar{\mathbf{C}}_k$  投影至世界坐标系下得到  $\mathbf{C}_k^w$ ，并加入到历史地图  $\mathbf{Map}$  之中。

### 3.2.4 回环检测

对于优化，我们采用通用图优化框架对一个闭环内的位姿进行调整。首先根据历史位姿建立节点。设经过位姿融合得到位姿序列  $\{\mathbf{Pose}'_i | i = 1, 2, \dots, M\}$ ，由于初始位姿序列频率较高为 10Hz，为了减小优化时的计算复杂度我们仅对关键帧作优化。我们将相对前一关键帧位移超过 5m 或者任一轴旋转角度超过  $30^\circ$  的位姿用作新的关键帧，从而获得关键帧位姿序列。将这些位姿设为位姿图结构的节点，用  $\{\mathbf{X}_i | i = 1, 2, \dots, N\}$  表示，其中  $\mathbf{X}_i$  为位姿变换矩阵，表示第  $i$  个关键帧的位姿。

其次在相互存在点云重叠的节点之间建立边的约束。用  $\{z_{ij} | (i, j) \in \Psi\}$  表示，其中  $\Psi$  表示存在点云重叠的关键帧二元组集合， $z_{ij}$  代表第  $i$  个关键帧与第  $j$  个关键帧点云之间的位姿变换观测值，我们采用 ICP 或者 NDT 算法进行点云配准从而计算  $z_{ij}$ 。由位姿  $\mathbf{X}_i$  与  $\mathbf{X}_j$  可计算第  $i$  个关键帧与第  $j$  个关键帧之间的位姿变换推算值，然而由于  $\mathbf{X}_i$  与  $\mathbf{X}_j$  并不一定准确，因此该推算值与观测值  $z_{ij}$  之间必然存在误差，记作  $e(\mathbf{X}_i, \mathbf{X}_j, z_{ij})$ ，由于  $z_{ij}$  是已知的，为表

述方便，我们将误差项写成如下形式：

$$e(X_i, X_j, z_{ij}) \triangleq e(X_i, X_j)$$

图结构中所有存在边连接的结点均可计算误差，从而组成误差平方函数：

$$F(X) = \sum_{(i,j) \in \Psi} e_{ij}(X)^T \Omega_{ij} e_{ij}(X)$$

式中的  $\Omega_{ij}$  称为信息矩阵，表示  $e_{ij}(X)$  中每个分量的权重。事实上信息矩阵是协方差矩阵的逆矩阵，最简单的情况下可以将其设为对角矩阵，对角元素的大小代表了我们对该误差项的重视程度。

设位姿图中存在  $n$  条边，优化的目标函数就是  $n$  条边所代表的误差矩阵平方和最小。

综上所述，使用位姿图进行位姿优化的过程为：

- (1) 按照位移超过 5m 或者任一旋转角超过  $30^\circ$  的标准抽取关键帧；
- (2) 将关键帧对应的位姿作为图节点，记作  $X$ ；
- (3) 对图的每一个节点，寻找与之具有较高点云重合度的节点，使用 ICP 的方法计算两个节点之间的位姿变换观测值作为节点之间的边；同时计算位姿变换观测值与位姿变换推测值的误差，组成误差平方函数  $F(X)$ ；
- (4) 采用 LM 的方法计算  $X$  的增量  $\Delta X$ ，并更新  $X$ ；
- (5) 重复迭代步骤(4)，直到  $F(X)$  取得极小值，此时所有节点位姿均得到优化调整。

基于几何位置的回环检测模块的缺陷：一是相隔稍远的关键帧之间点云重合度太小从而无法建立边的约束，二是 ICP 配准时陷入局部最优的情况时有发生。这两个缺陷影响了闭环优化的成功率。