深蓝学院
shenlanxueyuan.com

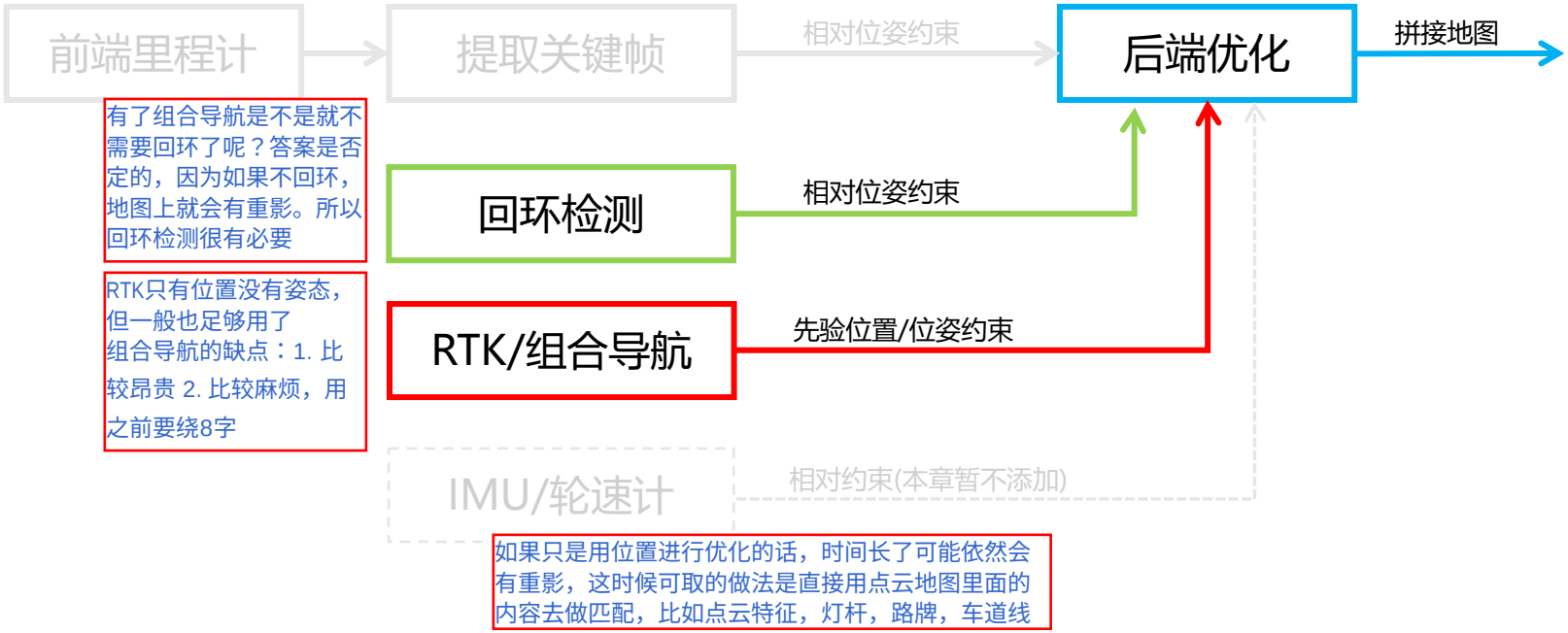
多传感器融合定位

第2章 基于地图的定位

主讲人 任 乾

北京理工大学本硕
自动驾驶从业者

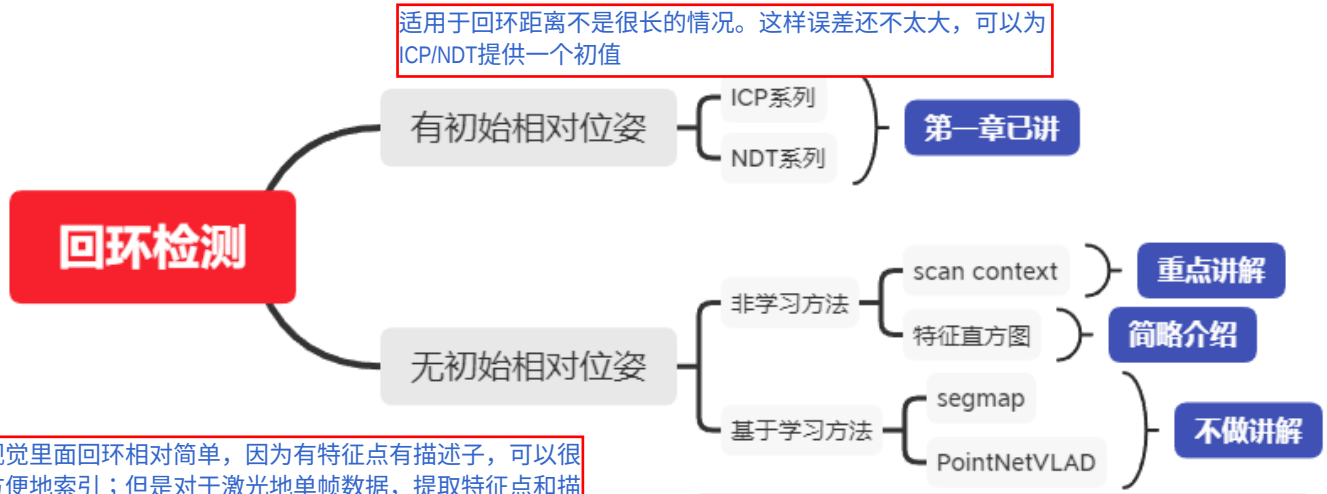
目录


前端里程计
提取关键帧
相对位姿约束
后端优化
拼接地图
有了组合导航是不是就不需要回环了呢？答案是否定的，因为如果不回环，地图上就会有重影。所以回环检测很有必要
回环检测
相对位姿约束
RTK只有位置没有姿态，但一般也足够用了
组合导航的缺点：1. 比较昂贵 2. 比较麻烦，用之前要绕8字
RTK/组合导航
先验位置/位姿约束
IMU/轮速计
相对约束(本章暂不添加)
如果只是用位置进行优化的话，时间长了可能依然会有重影，这时候可取的做法是直接用点云地图里面的内容去做匹配，比如点云特征，灯杆，路牌，车道线

目录

只用前端里程计拼成地图，一致性差很多
前端+回环之后，可以消除一部分误差。效果不太好，歪歪扭扭
前端+回环+先验信息之后，效果还不错

回环检测


回环检测
有初始相对位姿
ICP系列
NDT系列
第一章已讲
适用于回环距离不是很长的情况。这样误差还不太大，可以为ICP/NDT提供一个初值
无初始相对位姿
非学习方法
scan context
重点讲解
特征直方图
简略介绍
基于学习方法
segmap
PointNetVLAD
不做讲解
视觉里面回环相对简单，因为有特征点有描述子，可以很方便地索引；但是对于激光地单帧数据，提取特征点和描述子就没那么方便了。这里讲的两种方法都是在实际使用中经过检验或取得良好效果的。
一个是scan context，这个作者本人也在lego-loam等地方验证过；另一个是基于特征直方图，思路是划分栅格提取特征，根据特征相似程度来匹配
基于学习的方法不用优先考虑，因为两方面：
一方面它会使SLAM变得很重，另一方面它会对硬件提出很高的要求

问题：三维点云的回环检测计算量大且依赖初值
思路：可否将三维降成两维？

回环检测

基于Scan Context

切割点云

↓

生成scan context

↓

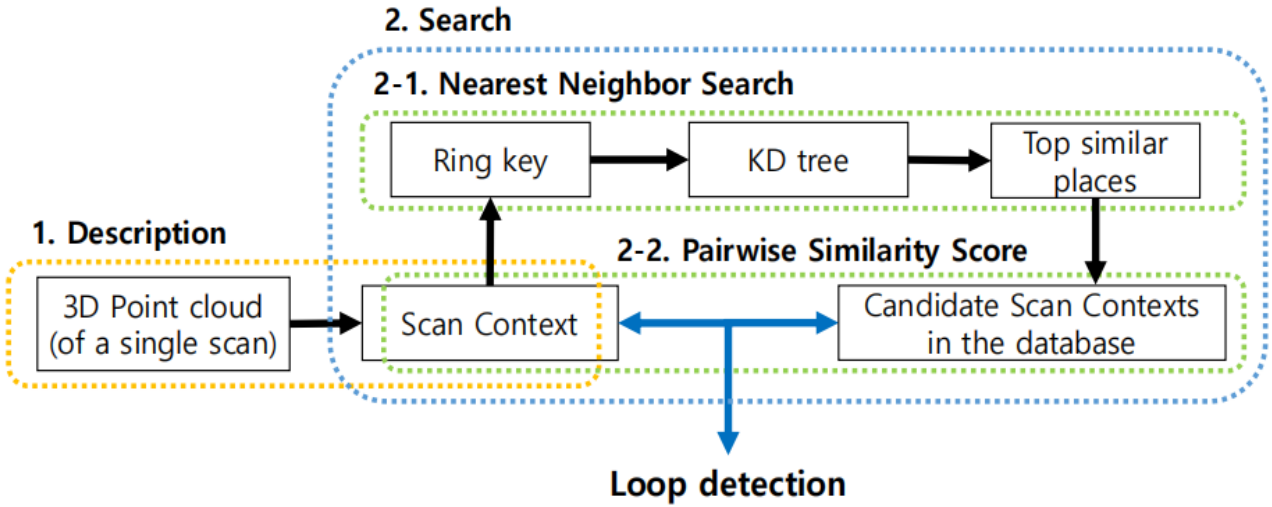
初步查找
(向量最近邻搜索)

↓

精确查找
(scan 相似度评分)

↓

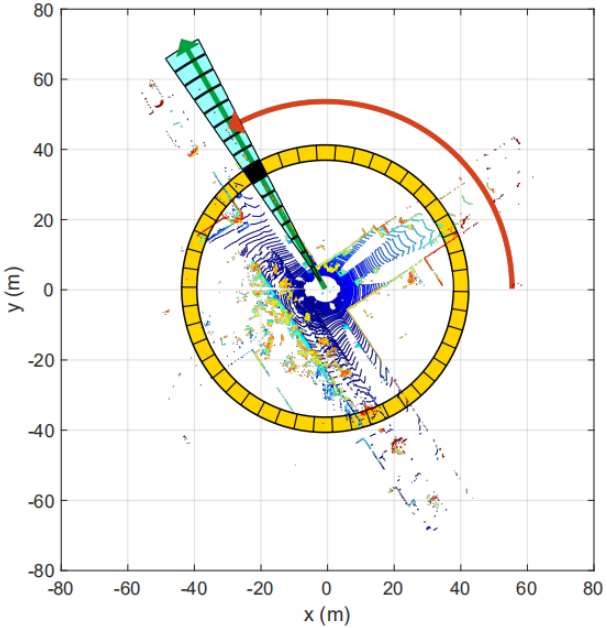
计算相对位姿



回环检测

基于Scan Context

1) 点云切割



点云分割示意图

a) 沿半径增大方向（绿色箭头方向），把点云空间等分成N_r个圆环。

每个圆环宽度为：

$$d_r = \frac{L_{\max}}{N_r}$$

其中 $L_{\max}$为激光点的最远距离。

b) 把每个圆环切割成N_s等份

分割后，激光点集合$\mathcal{P}$可重新表示为：

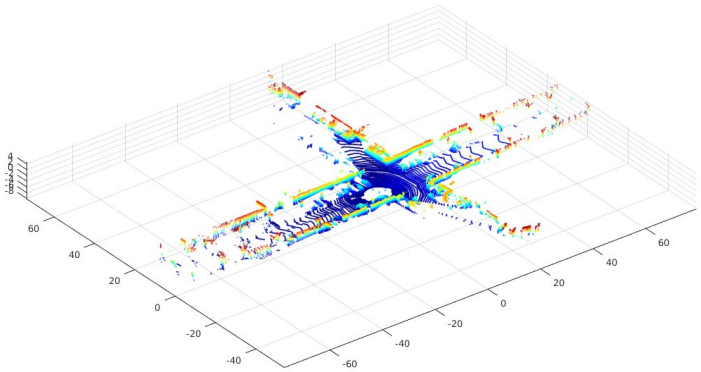
$$\mathcal{P} = \bigcup_{i\in[N_r],j\in[N_s]} \mathcal{P}_{ij}$$

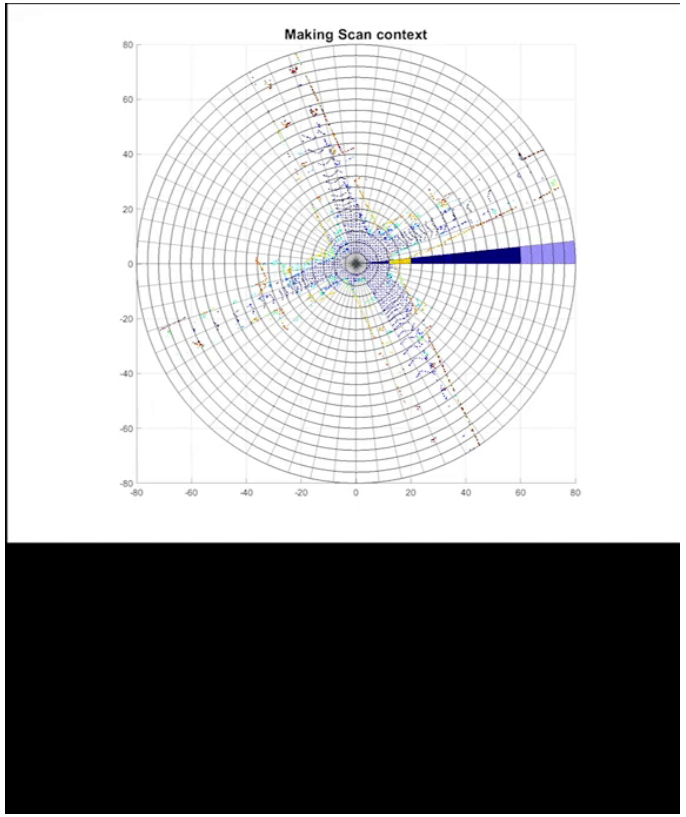
其中，$\mathcal{P}_{ij}$表示第i个圆环第j个扇形的分割单元中点的集合。

回环检测

基于Scan Context

2) 生成scan context

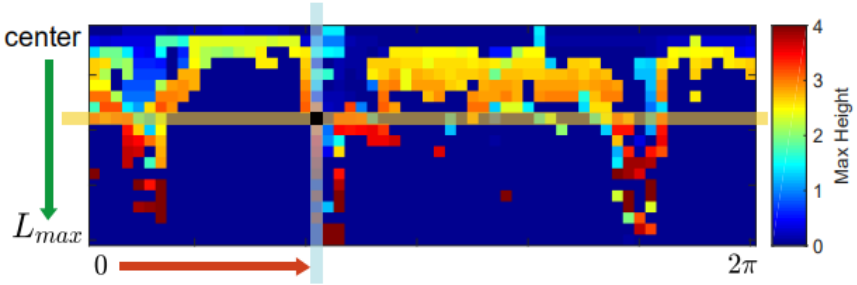




回环检测

基于Scan Context

2) 生成scan context



scan context示意图

把分割后的点云对应到$N_r \times N_s$的矩阵I(scan context)中：

a) 矩阵的每一行代表一个圆环；

b) 矩阵的每一列代表一个扇形；

c) 矩阵中每个元素的值代表该分割单元$\mathcal{P}_{ij}$中所有三维点的高度最大值。

回环检测

基于Scan Context

3) 基于scan context的匹配

I^q 为当前帧的scan context　　c_j^q 为 I^q中的第j列

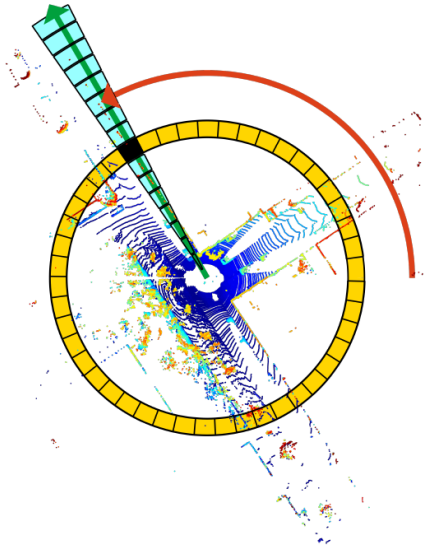
I^c 为历史帧的scan context　　c_j^c 为 I^c中的第j列

两帧scan context的距离函数定义为：

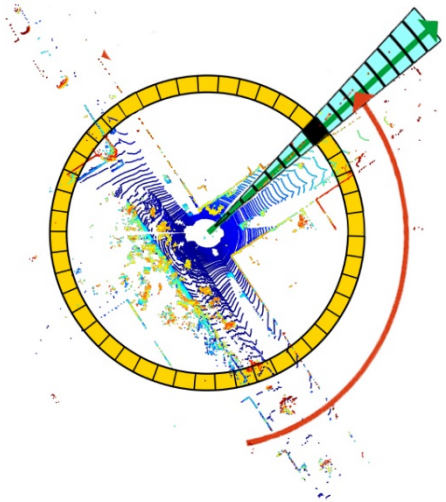
$$d\left(I^q, I^c\right) = \frac{1}{N_s}\sum_{j=1}^{N_s}\left(1 - \frac{c_j^q \cdot c_j^c}{\left\|c_j^q\right\|\left\|c_j^c\right\|}\right)$$

两帧scan context间的所有对应的列向量之间越相似，说明两帧点云越相似。距离函数小于某个阈值时，认为该历史帧为回环检测的回环帧。

问题：若当前帧相对于历史帧有旋转，比如同一个地方激光雷达经过时的方向相反，或者该地方为一个十字路口，激光雷达从不同方向经过（如右图所示），此时得到的scan context中列向量的顺序会发生改变，进而导致两帧的距离函数比较大。



当前帧的点云分割



历史帧的点云分割

回环检测

基于Scan Context

3) 基于scan context的匹配

I^q 为当前帧的scan context

I^c 为历史帧的scan context

方法：

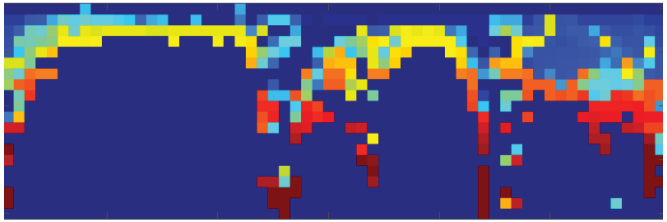
$$D\left(I^q, I^c\right) = \min_{n \in [N_s]} d\left(I^q, I_n^c\right)$$

将历史帧I^c按列平移，得到$[N_s]$个scan context，依次与当前帧的scan context计算距离，选择距离最小的那个。

根据以上方法，找出所有候选相似帧中和当前帧距离最小的，即为闭环匹配的帧。

匹配方法过于暴力……
所以实际操作的时候把每一列进行加和，采取了一个降维操作

I^q



I^c

I_n^c

回环检测

基于Scan Context

4) 计算相对位姿

假设距离最小时，对应的列的平移量为：

$$n^* = \underset{n \in [N_s]}{\operatorname{argmin}} d\left(I^q, I_n^c\right)$$

则它代表两帧之间有旋转，旋转的角度为：

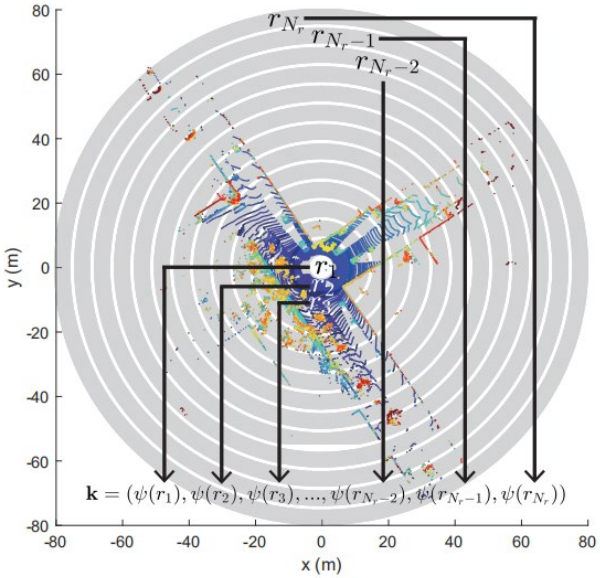
$$\phi = \frac{2\pi}{N_s} * n^*$$

以上旋转量则可作为ICP或NDT匹配的初始位姿，用于精确匹配得到闭环约束的相对位姿。

回环检测

基于Scan Context

5) 解决时间复杂度问题



目的：scan context可以用矩阵对应列的相似度来计算两帧的相似性，但是遍历所有历史帧的相似度计算量较高，需要做一个快速初步筛选。

思路：相似帧之间，落在同等半径的圆环中点的数量应相似，可用来快速查找。

方法：

a) 每帧生成一个向量：

$$\mathbf{k} = (\psi(r_1), \ldots, \psi(r_{N_r})), \text{ where } \psi : r_i \to \mathbb{R}$$

其中

$$\psi(r_i) = \frac{\|r_i\|_0}{N_s}$$

$\|r_i\|_0$ 表示半径 r_i 对应的圆环中非空分割单元的个数。

b) 根据a)中计算的向量，所有历史帧共同构建KDTree；

c) 使用当前帧对应的向量，在KDTree中查找，找出n个可能的相似帧，再通过scan context精确查找。

回环检测

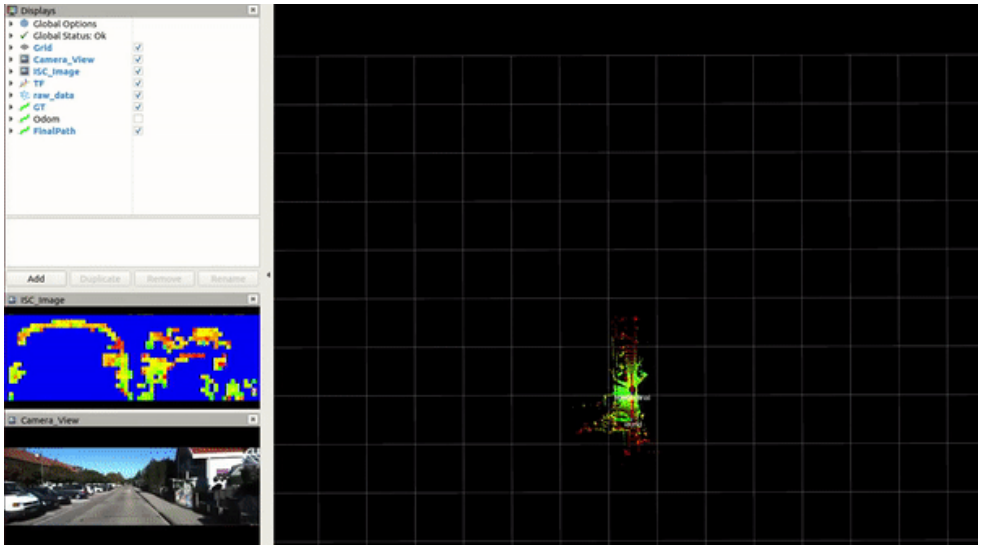
基于Scan Context

案例：

方法： LeGO-LOAM + scan context 闭环检测

代码：https://github.com/irapkaist/SC-LeGO-LOAM

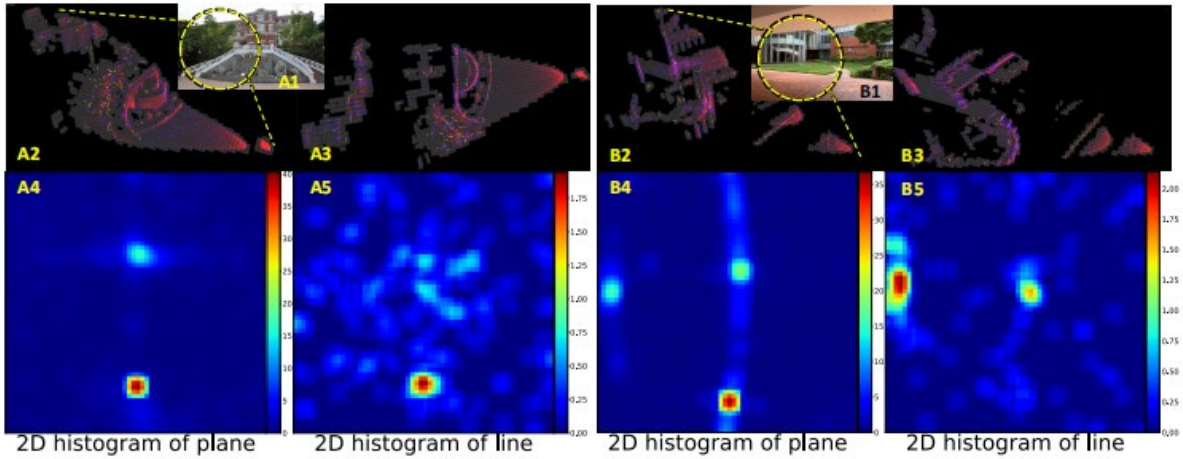
LeGO-LOAM相比LOAM有一个很大的优势，即它在存地图的时候，依然保存了帧与帧之间的约束关系，并没有因为地图拼接而丧失掉，这样我们就可以对历史状态做修正，这也是做闭环的一个前提条件。



回环检测

基于直方图

论文题目：A fast, complete, point cloud based loop closure for LiDAR odometryand mapping
应用案例：loam_livox
开源代码：https://github.com/hku-mars/loam_livox



目录

后端优化

1. 后端优化基本原理

目的：利用回环检测结果和惯导先验位姿修正里程计误差。

回环在此处提供的是两帧之间的相对位姿。

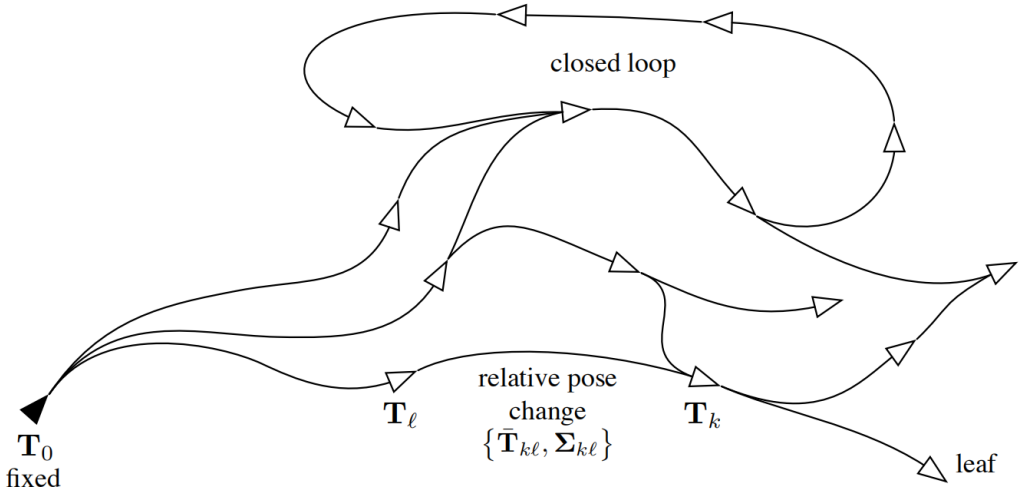
观测：

1) 连续两帧的相对位姿观测
2) 闭环匹配得到的相对位姿观测
3) 组合导航提供的先验位姿观测

1)和2)的观测构成了基于回环的位姿修正

1)和3)的观测构成了基于先验观测的位姿修正

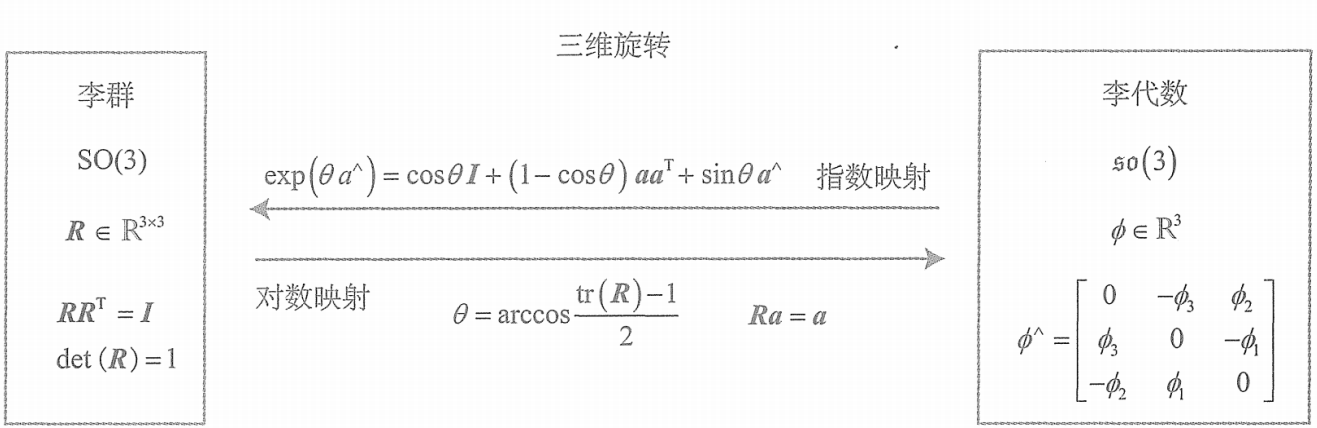
当然1) 2) 3) 也可以同时使用。



后端优化示意

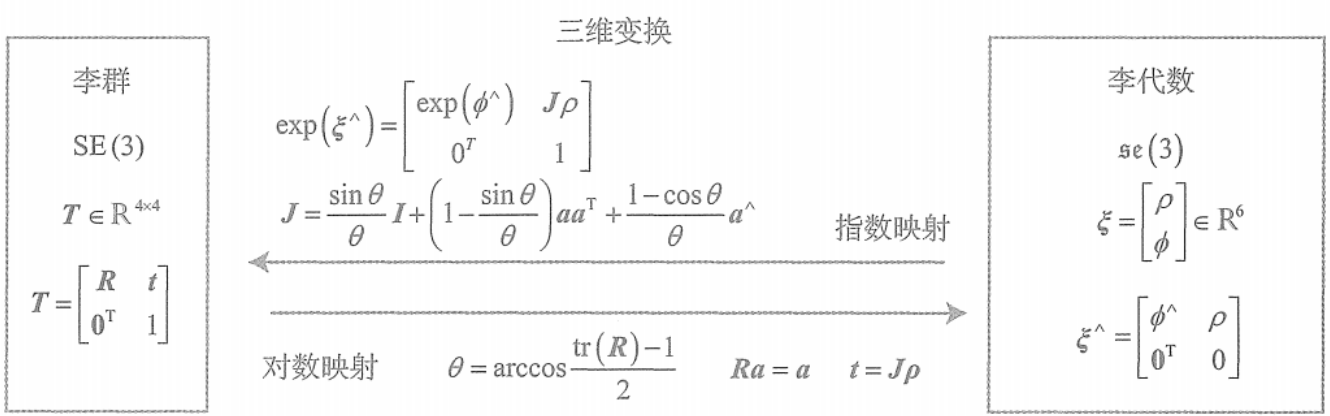
2. 李群、李代数基本知识

三维旋转上的定义：



2. 李群、李代数基本知识

三维变换上的定义：



2. 李群、李代数基本知识

BCH公式

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

SO(3)下的李括号定义为

$$[\phi_1, \phi_2] = (\Phi_1\Phi_2 - \Phi_2\Phi_1)^\vee$$

SE(3)下的李括号定义为

$$[\xi_1, \xi_2] = (\xi_1^\wedge\xi_2^\wedge - \xi_2^\wedge\xi_1^\wedge)^\vee$$

其中

$$\Phi = \phi^\wedge = \begin{bmatrix} 0 & -\phi(3) & \phi(2) \\ \phi(3) & 0 & -\phi(1) \\ -\phi(2) & \phi(1) & 0 \end{bmatrix}$$

2. 李群、李代数基本知识

SO(3)对应的BCH公式

$$\ln\left(\exp\left(\phi_1^{\wedge}\right)\exp\left(\phi_2^{\wedge}\right)\right)^{\vee} \approx \begin{cases} J_l\left(\phi_2\right)^{-1}\phi_1+\phi_2 & \text{，当 } \phi_1 \text{ 为小量} \\ J_r\left(\phi_1\right)^{-1}\phi_2+\phi_1 & \text{，当 } \phi_2 \text{ 为小量} \end{cases}$$

其中左乘雅可比为

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

即

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

右乘雅可比仅需要在左乘雅可比的基础上对自变量取负号，即

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

2. 李群、李代数基本知识

SE(3)对应的BCH公式

$$\ln\left(\exp\left(\boldsymbol{\xi}_1^{\wedge}\right)\exp\left(\boldsymbol{\xi}_2^{\wedge}\right)\right)^{\vee} \approx \begin{cases} \mathcal{J}_{\ell}\left(\boldsymbol{\xi}_2\right)^{-1}\boldsymbol{\xi}_1+\boldsymbol{\xi}_2\text{，当 }\boldsymbol{\xi}_1\text{ 为小量} \\ \mathcal{J}_{r}\left(\boldsymbol{\xi}_1\right)^{-1}\boldsymbol{\xi}_2+\boldsymbol{\xi}_1\text{，当 }\boldsymbol{\xi}_2\text{ 为小量} \end{cases}$$

由于SE(3)上的雅可比形式过于复杂，此处直接给出本章所用到的近似形式如下。详细内容可参考《机器人中的状态估计》中公式7.83的推导过程。

$$\mathcal{J}_r^{-1}(\boldsymbol{\xi}) \approx I+\frac{1}{2}\begin{bmatrix} \boldsymbol{\phi}^{\wedge} & \boldsymbol{\rho}^{\wedge} \\ 0 & \boldsymbol{\phi}^{\wedge} \end{bmatrix}$$

若 $\boldsymbol{\xi}$ 非常小，则该雅可比可以直接使用单位阵，此时有

$$\ln\left(\exp\left(\boldsymbol{\xi}_1^{\wedge}\right)\exp\left(\boldsymbol{\xi}_2^{\wedge}\right)\right)^{\vee} \approx \ln\left(\exp\left(\boldsymbol{\xi}_1^{\wedge}+\boldsymbol{\xi}_1^{\wedge}\right)\right)^{\vee}$$

这个地方值得注意

2. 李群、李代数基本知识

伴随的作用：
在进行公式推到时进行变量转移

SO(3)上的伴随性质

$$R\exp\left(p^{\wedge}\right)R^{\mathrm{T}}=\exp\left((Rp)^{\wedge}\right)$$

SE(3)上的伴随性质

$$T\exp\left(\xi^{\wedge}\right)T^{-1}=\exp\left((\mathrm{Ad}(T)\xi)^{\wedge}\right)$$

其中伴随矩阵的定义如下

$$\mathrm{Ad}(T)=\begin{bmatrix}R & t^{\wedge}R\\ 0 & R\end{bmatrix}$$

后端优化

3. 基于回环的位姿修正

位姿图优化是把所有的观测和状态放在一起优化，残差项是前面所讲残差项的总和。

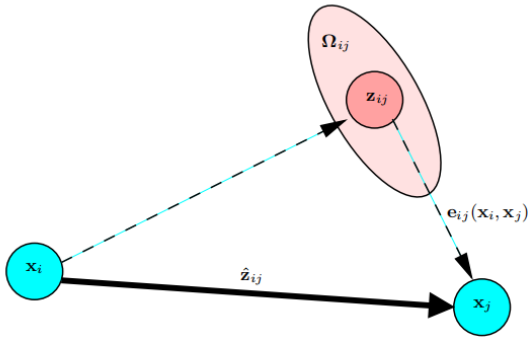
在实际使用中，各残差会被分配一个权重，也就是信息矩阵，它相当于对残差进行加权。

考虑信息矩阵后，总的残差项可以表示为：

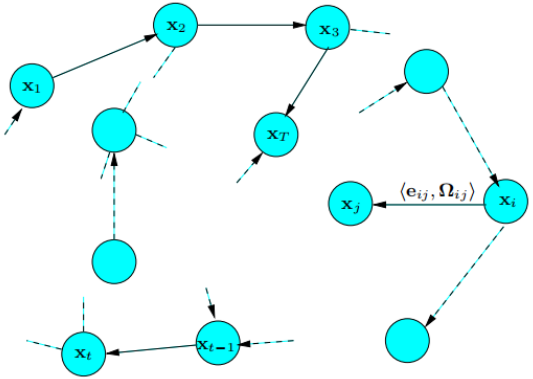
$$\mathbf{F}(\mathbf{x}) = \sum_{\langle i,j\rangle \in \mathcal{C}} \underbrace{\mathbf{e}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{e}_{ij}}_{\mathbf{F}_{ij}}$$

此时优化问题可以表示为：

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \mathbf{F}(\mathbf{x})$$



单个残差项



多个残差项组成位姿图

后端优化

3. 基于回环的位姿修正

第i和j帧之间的观测，在李群SE(3)上可以表示为

$$\boldsymbol{\Delta T}_{ij} = \boldsymbol{T}_i^{-1}\boldsymbol{T}_j$$

也可以在李代数上表示为

$$\Delta\boldsymbol{\xi}_{ij} = \boldsymbol{\xi}_i^{-1} \circ \boldsymbol{\xi}_j = \ln\left(\boldsymbol{T}_i^{-1}\boldsymbol{T}_j\right)^{\vee}$$

若位姿没有误差，则上面两个式子是精确相等的，但当位姿有误差存在时，便可以使用等式的左右两端计算残差项。

$$\begin{aligned}\boldsymbol{e}_{ij} &= \ln\left(\Delta\boldsymbol{T}_{ij}^{-1}\boldsymbol{T}_i^{-1}\boldsymbol{T}_j\right)^{\vee} \\ &= \ln\left(\exp\left(\left(-\boldsymbol{\xi}_{ij}\right)^{\wedge}\right)\exp\left(\left(-\boldsymbol{\xi}_i\right)^{\wedge}\right)\exp\left(\boldsymbol{\xi}_j^{\wedge}\right)\right)^{\vee}\end{aligned}$$

位姿图优化的思想是通过调整状态量(即位姿)，使残差项的值最小化，这就需要用残差项对位姿求雅可比，才能使用梯度下降方法进行优化。

求雅可比的方式是对位姿添加扰动，此时残差表示为：

$$\hat{\boldsymbol{e}}_{ij} = \ln\left(\boldsymbol{T}_{ij}^{-1}\boldsymbol{T}_i^{-1}\exp\left(\left(-\boldsymbol{\delta\xi}_i\right)^{\wedge}\right)\exp\left(\delta\boldsymbol{\xi}_j^{\wedge}\right)\boldsymbol{T}_j\right)^{\vee}$$

后端优化

3. 基于回环的位姿修正

进一步对前面的式子进行化简

$$\hat{\boldsymbol{e}}_{ij} = \ln\left(\boldsymbol{T}_{ij}^{-1}\boldsymbol{T}_i^{-1}\exp\left(\left(-\boldsymbol{\delta\xi}_i\right)^{\wedge}\right)\exp\left(\delta\boldsymbol{\xi}_j^{\wedge}\right)\boldsymbol{T}_j\right)^{\vee}$$

伴随性质

$$= \ln\left(\boldsymbol{T}_{ij}^{-1}\boldsymbol{T}_i^{-1}\boldsymbol{T}_j\exp\left(\left(-\operatorname{Ad}\left(\boldsymbol{T}_j^{-1}\right)\boldsymbol{\delta\xi}_i\right)^{\wedge}\right)\exp\left(\left(\operatorname{Ad}\left(\boldsymbol{T}_j^{-1}\right)\boldsymbol{\delta\xi}_j\right)^{\wedge}\right)\right)^{\vee}$$

小量情形下可合并

$$\approx \ln\left(\exp(\boldsymbol{e}_{ij})\exp\left(\left(-\operatorname{Ad}\left(\boldsymbol{T}_j^{-1}\right)\boldsymbol{\delta\xi}_i\right)^{\wedge}+\left(\operatorname{Ad}\left(\boldsymbol{T}_j^{-1}\right)\boldsymbol{\delta\xi}_j\right)^{\wedge}\right)\right)^{\vee}$$

$$\approx \boldsymbol{e}_{ij} - \mathcal{J}_r^{-1}\left(e_{ij}\right)\operatorname{Ad}\left(\boldsymbol{T}_j^{-1}\right)\boldsymbol{\delta\xi}_i + \mathcal{J}_r^{-1}\left(e_{ij}\right)\operatorname{Ad}\left(\boldsymbol{T}_j^{-1}\right)\boldsymbol{\delta\xi}_j$$

上面的式子表明，残差关于T_i的雅克比为

$$A_{ij} = \frac{\partial \boldsymbol{e}_{ij}}{\partial \boldsymbol{\delta\xi}_i} = -\mathcal{J}_r^{-1}\left(\boldsymbol{e}_{ij}\right)\operatorname{Ad}\left(\boldsymbol{T}_j^{-1}\right)$$

残差关于T_j的雅克比为

$$B_{ij} = \frac{\partial \boldsymbol{e}_{ij}}{\partial \boldsymbol{\delta\xi}_j} = \mathcal{J}_r^{-1}\left(\boldsymbol{e}_{ij}\right)\operatorname{Ad}\left(\boldsymbol{T}_j^{-1}\right)$$

其中

$$\mathcal{J}_r^{-1}\left(\boldsymbol{e}_{ij}\right) \approx I + \frac{1}{2}\begin{bmatrix}\phi_e^{\wedge} & \rho_e^{\wedge} \\ 0 & \phi_e^{\wedge}\end{bmatrix}$$

后端优化

3. 基于回环的位姿修正

为了找到梯度方向，需要对残差进行一阶泰勒展开

$$
\begin{aligned}
&\mathbf{e}_{ij}\left(\mathbf{x}_i+\Delta\mathbf{x}_i,\mathbf{x}_j+\Delta\mathbf{x}_j\right)\\
&=\mathbf{e}_{ij}(\mathbf{x}+\Delta\mathbf{x})\\
&\approx\mathbf{e}_{ij}+\mathbf{J}_{ij}\Delta\mathbf{x}
\end{aligned}
$$

其中 J_{ij} 即为前面推导的残差关于位姿的雅可比组成的矩阵

$$
\mathbf{J}_{ij}=(0\cdots0\underbrace{\mathbf{A}_{ij}}_{\text{node }i}\mathbf{0}\cdots0\underbrace{\mathbf{B}_{ij}}_{\text{node }j}\mathbf{0}\cdots0)
$$

对于每一个残差块，便有

$$
\begin{aligned}
&\mathbf{F}_{ij}(\mathbf{x}+\Delta\mathbf{x})\\
&=\mathbf{e}_{ij}(\mathbf{x}+\Delta\mathbf{x})^T\Omega_{ij}\mathbf{e}_{ij}(\mathbf{x}+\Delta\mathbf{x})\\
&\approx\left(\mathbf{e}_{ij}+\mathbf{J}_{ij}\Delta\mathbf{x}\right)^T\Omega_{ij}\left(\mathbf{e}_{ij}+\mathbf{J}_{ij}\Delta\mathbf{x}\right)\\
&=\underbrace{\mathbf{e}_{ij}^T\Omega_{ij}\mathbf{e}_{ij}}_{c_{ij}}+2\underbrace{\mathbf{e}_{ij}^T\Omega_{ij}\mathbf{J}_{ij}}_{\mathbf{b}_{ij}^T}\Delta\mathbf{x}+\Delta\mathbf{x}^T\underbrace{\mathbf{J}_{ij}^T\Omega_{ij}\mathbf{J}_{ij}}_{\mathbf{H}_{ij}}\Delta\mathbf{x}\\
&=c_{ij}+2\mathbf{b}_{ij}^T\Delta\mathbf{x}+\Delta\mathbf{x}^T\mathbf{H}_{ij}\Delta\mathbf{x}
\end{aligned}
$$

其中

$$
\mathbf{H}_{ij}=\begin{pmatrix}
\ddots & & & \\
& \mathbf{A}_{ij}^T\mathbf{\Omega}_{ij}\mathbf{A}_{ij} & \cdots & \mathbf{A}_{ij}^T\mathbf{\Omega}_{ij}\mathbf{B}_{ij} \\
\vdots & & \ddots & \vdots \\
& \mathbf{B}_{ij}^T\mathbf{\Omega}_{ij}\mathbf{A}_{ij} & \cdots & \mathbf{B}_{ij}^T\mathbf{\Omega}_{ij}\mathbf{B}_{ij} \\
& & & \ddots
\end{pmatrix}
$$

$$
\mathbf{b}_{ij}=\begin{pmatrix}
\vdots \\
\mathbf{A}_{ij}^T\mathbf{\Omega}_{ij}\mathbf{e}_{ij} \\
\vdots \\
\mathbf{B}_{ij}^T\mathbf{\Omega}_{ij}\mathbf{e}_{ij} \\
\vdots
\end{pmatrix}
$$

后端优化

3. 基于回环的位姿修正

因此总的残差项可以表示为

$$\begin{aligned}\mathbf{F}(\mathbf{x}+\Delta\mathbf{x}) &= \sum_{\langle i,j\rangle\in\mathcal{C}} \mathbf{F}_{ij}(\mathbf{x}+\Delta\mathbf{x}) \\ &\approx \sum_{\langle i,j\rangle\in\mathcal{C}} \left(\mathrm{c}_{ij} + 2\mathbf{b}_{ij}^T\Delta\mathbf{x} + \Delta\mathbf{x}^T\mathbf{H}_{ij}\Delta\mathbf{x}\right) \\ &= \mathrm{c} + 2\mathbf{b}^T\Delta\mathbf{x} + \Delta\mathbf{x}^T\mathbf{H}\Delta\mathbf{x}\end{aligned}$$

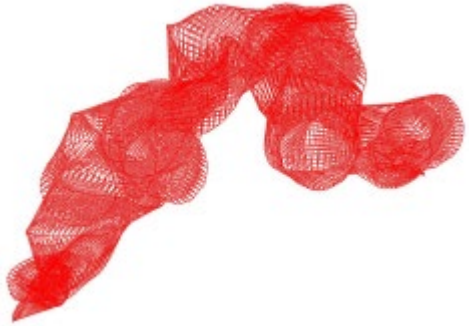
若要误差最小，只需使得

$$\mathbf{H}\Delta\mathbf{x} = -\mathbf{b}$$

根据修正量，修正X的值，即完成一次迭代

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

多次迭代，直至残差满足收敛条件时，则终止循环，完成优化。



优化前

优化后

后端优化

4. 基于先验观测的位姿修正

先验观测是一元边，它不像前面所述的帧间观测连接两个位姿状态，而是只连接一个位姿状态量，它直接给出的就是该状态量的观测值，因此它对应的残差就是观测值与状态量之间的差异，即

$$\begin{aligned} \boldsymbol{e}_i &= \ln\left(\boldsymbol{Z}_i^{-1}\boldsymbol{T}_i\right)^{\vee} \\ &= \ln\left(\exp\left(\left(-\boldsymbol{\xi}_{zi}\right)^{\wedge}\right)\exp\left(\boldsymbol{\xi}_i^{\wedge}\right)\right)^{\vee} \end{aligned}$$

对残差添加扰动，可得

$$\hat{\boldsymbol{e}}_i = \ln\left(\boldsymbol{Z}_i^{-1}\exp\left(\delta\boldsymbol{\xi}_i^{\wedge}\right)\boldsymbol{T}_i\right)^{\vee}$$

利用伴随性质和BCH公式进行化简，可得

$$\begin{aligned} \hat{\boldsymbol{e}}_i &= \ln\left(\boldsymbol{Z}_i^{-1}\boldsymbol{T}_i\exp\left(\left(\mathrm{Ad}\left(\boldsymbol{T}_i^{-1}\right)\boldsymbol{\delta\xi}_i\right)^{\wedge}\right)\right)^{\vee} \\ &= \ln\left(\exp\left(\boldsymbol{e}_i\right)\exp\left(\left(\mathrm{Ad}\left(\boldsymbol{T}_i^{-1}\right)\boldsymbol{\delta\xi}_i\right)^{\wedge}\right)\right)^{\vee} \\ &\approx e_i + \mathcal{J}_r^{-1}\left(\boldsymbol{e}_i\right)\mathrm{Ad}\left(\boldsymbol{T}_i^{-1}\right)\boldsymbol{\delta\xi}_i \end{aligned}$$

因此，残差关于T_i的雅可比为

$$\frac{\partial \boldsymbol{e}_i}{\partial \boldsymbol{\delta\xi}_i} = \mathcal{J}_r^{-1}\left(\boldsymbol{e}_i\right)\mathrm{Ad}\left(\boldsymbol{T}_i^{-1}\right)$$

其中

$$\mathcal{J}_r^{-1}\left(e_i\right) \approx I + \frac{1}{2}\begin{bmatrix} \phi_e^{\wedge} & \rho_e^{\wedge} \\ 0 & \phi_e^{\wedge} \end{bmatrix}$$

后面的推导过程，便与相对位姿做观测的推导过程完全一致。

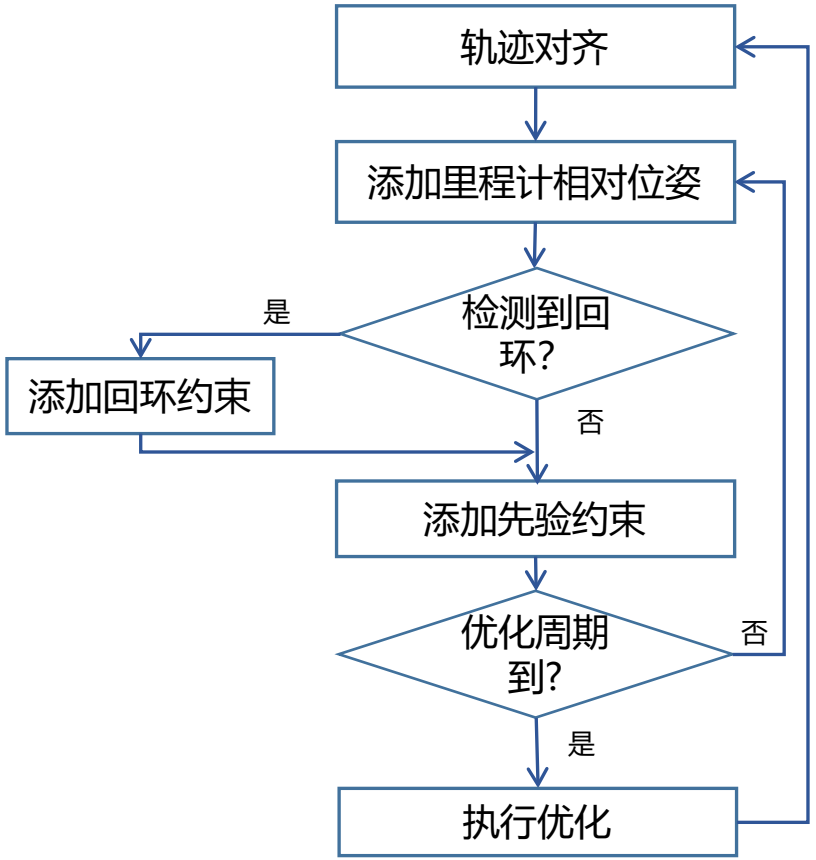
此外，部分场合提供的观测只有位置，没有姿态，比如只有RTK，而没有组合导航，这里的残差便只剩下位置误差。相应的雅可比公式，可自行推导。

目录

点云地图建立

1. 整体流程

建图流程设计的核心原则是准确、高效地把里程计相对位姿、回环相对位姿、惯导先验位姿进行融合。



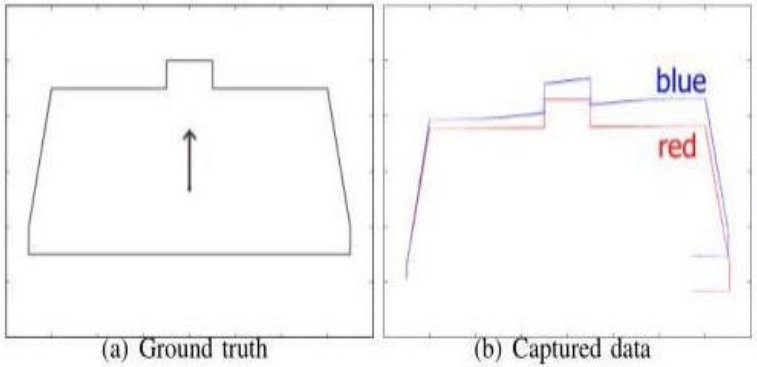
点云地图建立流程图

点云地图建立

2. 畸变补偿

1) 产生原因

一帧点云中的激光点是不同时刻采集的，激光点的坐标原点是采集时刻的雷达位姿。雷达在不同时刻的位姿有变化时，各激光点原点不一致，拼接成一帧时，点云的形状便和实际物体形状不一致。



2) 补偿方法

对每个激光点坐标做补偿，补偿量为激光点原点(即当时雷达坐标)相对于该帧起始时刻的变化。

点云地图建立

2. 畸变补偿

3) 补偿公式

假设一帧点云中，起始时刻雷达的位姿为

$$T_0 = \begin{bmatrix} R_0 & t_0 \\ 0 & 1 \end{bmatrix}$$

第i个激光点采集时，雷达的位姿为

$$T_i = \begin{bmatrix} R_i & t_i \\ 0 & 1 \end{bmatrix}$$

第i个激光点的坐标为

$$P_i = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} \end{bmatrix}^T$$

则第i个激光点补偿畸变后的坐标应该为

$$\bar{P}_i = T_0^{-1} T_i P_i$$

上式可以理解为，只需要计算0到i时刻，激光雷达的相对旋转和相对平移变化即可。

实际上，雷达点云是局部坐标系下的表示，当以0时刻雷达的位姿为基准坐标系时，此时T_0即为单位阵，T_i即为0到i时刻的相对旋转和平移。

此时有

$$R_i = w \bigtriangledown t$$
$$t_i = V \bigtriangledown t$$

即，只需要知道 0 到 i 时刻的平均角速度和平均速度即可。

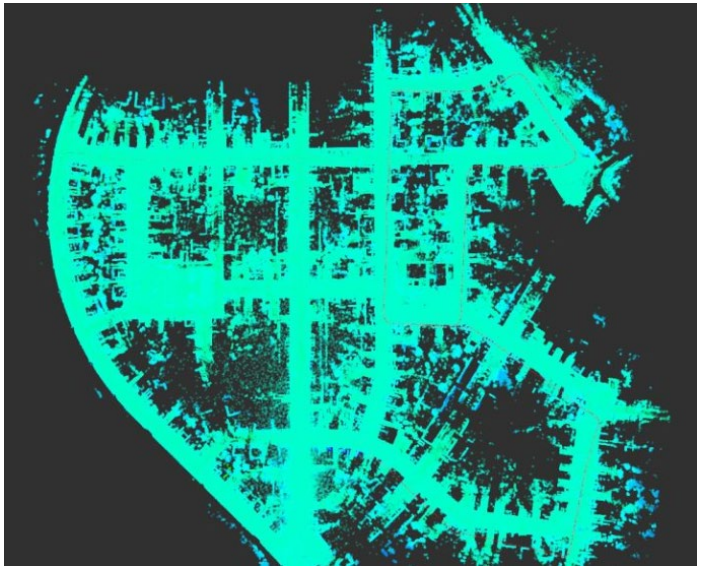
点云地图建立

3. 建图流程代码讲解

基于g2o的代码实现。

对照代码框架，讲解回环约束和先验观测的添加流程。

最终效果建图

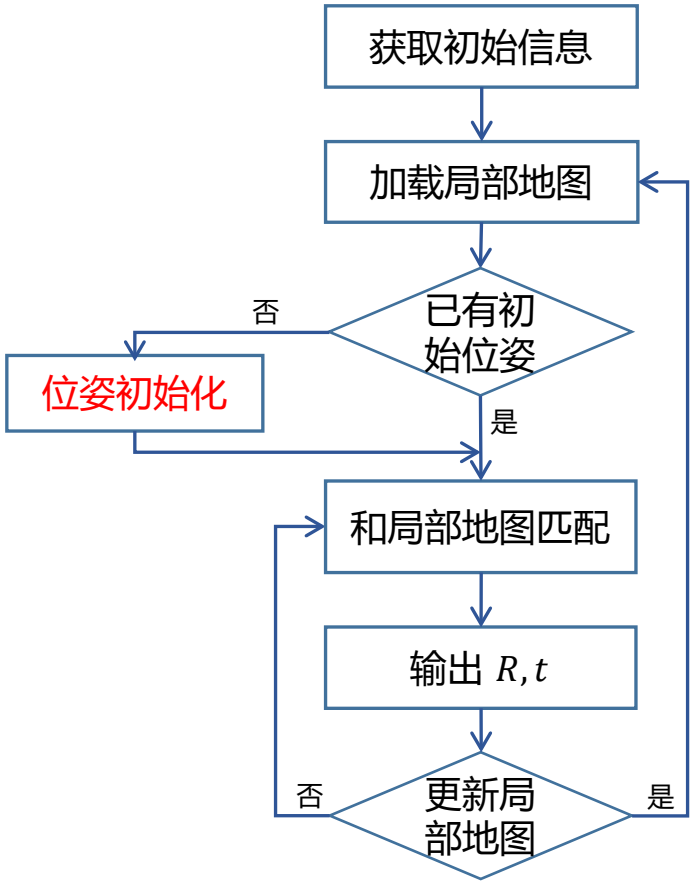


目录

基于地图的定位

1. 整体流程

在地图匹配中，鲁棒性和运行速度更加重要，因此实际使用中，基于NDT的匹配使用更广泛。



基于地图定位流程图

基于地图的定位

2. 位姿初始化

由于NDT匹配需要较准确的初始位姿，因此在定位之前需要初始化环节，给出载体的初始位姿。

按照难度由低到高，常见的初始化需求有这样几种：

1）已知位姿的初始化

没有双天线时，惯导初始化需要绕8字

2）位置已知而姿态位置的初始化

可以用磁力计

3）位置和姿态均未知的初始化

作业

1. 闭环修正及精度评价

提供的工程框架中已经给出了闭环的流程和实现，但是是基于ICP的，这是在已知粗略位姿情况下的实现方式。在未知粗略位姿情况下，闭环检测的实现难度会加大。

要求使用前面讲过的scan context，实现此功能。并和已有的图优化功能完成对接，实现修正。并最终给出修正前后的轨迹精度对比。

2. 位姿初始化

提供的工程框架中已经给出了位姿初始化功能，但是是在起始位置的，并且是基于已知粗略位姿的。

要求实现地图中任意位置的位姿初始化，可以从三种难度等级中任选一种，难度越高，得分越高。

深蓝学院
shenlanxueyuan.com

感谢聆听！

Thanks for Listening