深蓝学院
shenlanxueyuan.com

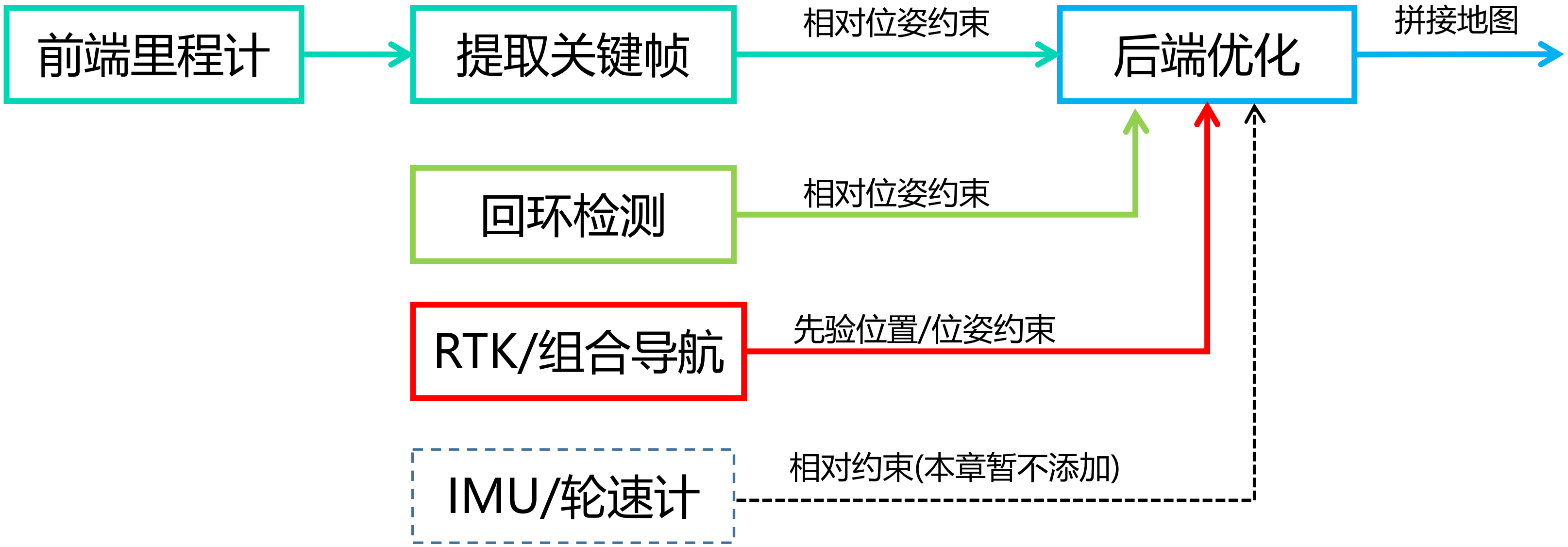
# 多传感器融合定位

## 第1讲 3D激光里程计

主讲人 任 乾
北京理工大学本硕
自动驾驶从业者

# 目录

# 整体流程介绍



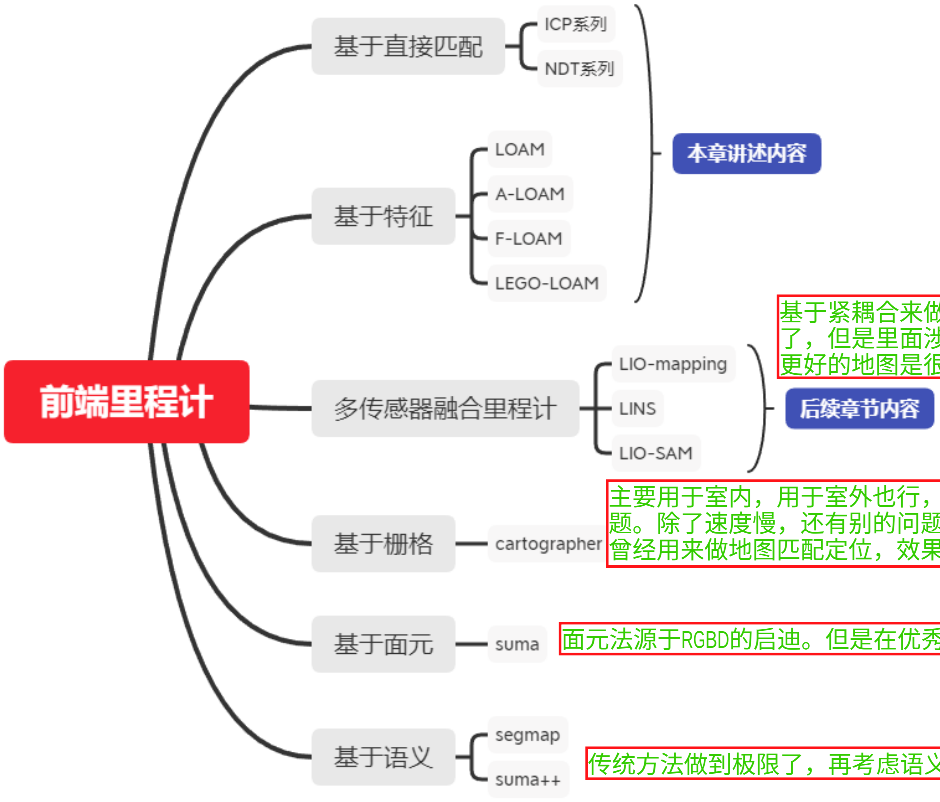

点云地图构建流程

# 整体流程介绍

前端里程计
提取关键帧
相对位姿约束
后端优化
拼接地图
回环检测
相对位姿约束
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)
点云地图构建流程

# 目录

# 前端里程计方案




基于紧耦合来做前端里程计可能太笨重了，但是里面涉及的一些技术对于建一个更好的地图是很有用的。

主要用于室内，用于室外也行，现在也支持融入RTK。但是大场景下有些问题。除了速度慢，还有别的问题。
曾经用来做地图匹配定位，效果不是特别好

面元法源于RGBD的启迪。但是在优秀工程里面用的很少

传统方法做到极限了，再考虑语义

# 前端里程计方案—基于直接匹配

## ICP系列—点到点ICP

接触任何技术上的东西，都要首先用白话讲清它的思路

点集：

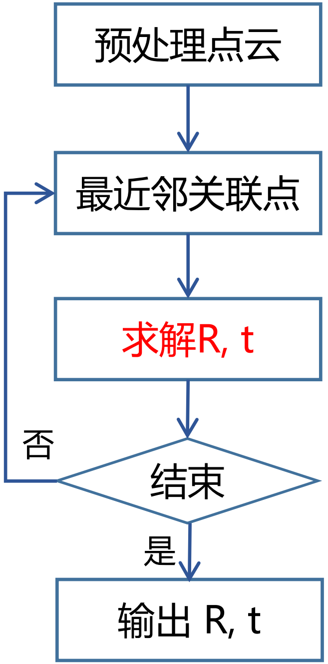
$$X = \{x_1, x_2, \cdots, x_{N_x}\}$$

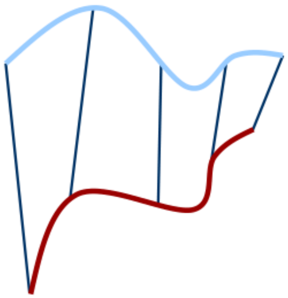
$$Y = \{y_1, y_2, \cdots, y_{N_y}\}$$

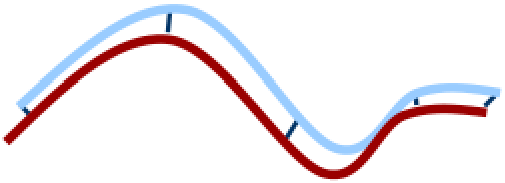
其中$X$和$Y$是原始点云的子集，选取的是两个点集中能够互相关联的那些点，即$N_x = N_y$

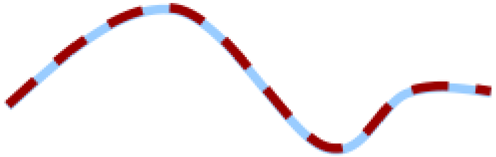
目标：

$$minE(R,t) = min\frac{1}{N_y}\sum_{i=1}^{N_y}\|x_i - Ry_i - t\|^2$$







# 前端里程计方案—基于直接匹配

## ICP系列—点到点ICP

基本思路：
通过去中心化，把平移和旋转解耦开!后面的所有过程都是这个思路的展开。

学习一个公式的推导，最关键的是理清它的思路，看出它最关键的一步或者几步是怎样设计的。

$$\begin{aligned}
E(R,t) &= \frac{1}{N_y}\sum_{i=1}^{N_y}\|x_i - Ry_i - t - u_x + Ru_y + u_x - Ru_y\|^2 \\
&= \frac{1}{N_y}\sum_{i=1}^{N_y}\left(\|x_i - u_x - R(y_i - u_y) + (u_x - Ru_y - t)\|^2\right) \\
&= \frac{1}{N_y}\sum_{i=1}^{N_y}\left(\|x_i - u_x - R(y_i - u_y)\|^2 + \|u_x - Ru_y - t\|^2 + 2\boxed{(x_i - u_x - R(y_i - u_y))}^T(u_x - Ru_y - t)\right) \\
&= \frac{1}{N_y}\sum_{i=1}^{N_y}\left(\|x_i - u_x - R(y_i - u_y)\|^2 + \|u_x - Ru_y - t\|^2\right)
\end{aligned}$$

由 $u_x$ 和 $u_y$ 的定义可知，
该项累加和为零

步骤 ：
1．去中心
2．展开并消除交叉项
3．旋转平移解耦

其中 $u_x$ 和 $u_y$ 分别是点集$X$和$Y$的质心，即

$$u_x = \frac{1}{N_x}\sum_{i=1}^{N_x} x_i$$

$$u_y = \frac{1}{N_y}\sum_{i=1}^{N_y} y_i$$

令 $E_1(R,t) = \frac{1}{N_y}\sum_{i=1}^{N_y}\|x_i - u_x - R(y_i - u_y)\|^2$

$E_2(R,t) = \|u_x - Ru_y - t\|^2$

那么，对于任意的$R$，均可以找到一个$t$，使得$u_x - Ru_y - t = 0$，即$E_2(R,t) = 0$

# 前端里程计方案—基于直接匹配

## ICP系列—点到点ICP

$$
\begin{aligned}
E_1(R,t) &= \frac{1}{N_y}\sum_{i=1}^{N_y}\|x_i - u_x - R\left(y_i - u_y\right)\|^2 \\
&= \frac{1}{N_y}\sum_{i=1}^{N_y}\|x_i' - Ry_i'\|^2 \\
&= \frac{1}{N_y}\sum_{i=1}^{N_y}\left(\boxed{x_i'^T x_i'} + y_i'^T \boxed{R^T R} y_i' - 2x_i'^T R y_i'\right)
\end{aligned}
$$

与R无关　　单位阵

令 $E_1'(R,t) = \sum_{i=1}^{N_y} x_i'^T R y_i'$

则 $minE_1(R,t)$

即 $maxE_1'(R,t)$

步骤 ：
4. 变符号，展开

# 前端里程计方案—基于直接匹配

## ICP系列—点到点ICP

$$E_1'(R,t)=\sum_{i=1}^{N_y} x_i'^T R y_i' \overset{(1)}{=} \sum_{i=1}^{N_y} \mathrm{Trace}\left(x_i'^T R y_i'\right) \overset{(2)}{=} \sum_{i=1}^{N_y} \mathrm{Trace}\left(R y_i' x_i'^T\right) = \mathrm{Trace}\left(\sum_{i=1}^{N_y} R y_i' x_i'^T\right) \overset{(3)}{=} \mathrm{Trace}(RH)$$

等式(1)：常数的迹等于它自身

等式(2)： $\mathrm{tr}(AB)=\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}b_{ji}=\sum_{i=1}^{m}\sum_{j=1}^{n} b_{ji}a_{ij}=\mathrm{tr}(BA)$

等式(3)： $H=\sum_{i=1}^{N_y} y_i' x_i'^T$

问题转化为，找到合适的$R$，使 $\mathrm{Trace}(RH)$ 达到最大值

## ICP系列—点到点ICP

定理：若有正定矩阵 $AA^T$，则对于任意正交矩阵B，有 $\mathrm{Trace}(AA^T) \geq \mathrm{Trace}(BAA^T)$

意义：若能找到$R$，把 $\mathrm{Trace}(RH)$ 转换成 $\mathrm{Trace}\left(AA^T\right)$ 的形式，则该$R$就是我们要找的旋转矩阵

证明：

$$\mathrm{tr}\left(BAA^T\right)=\mathrm{tr}\left(A^TBA\right)=\sum_i a_i^T\left(Ba_i\right)$$

其中 $a_i$ 为$A$的列向量。根据柯西-施瓦茨不等式，有

$$a_i^T\left(Ba_i\right) \leq \sqrt{\left(a_i^Ta_i\right)\left(a_i^TB^TBa_i\right)}=a_i^Ta_i$$

因此

$$\mathrm{tr}\left(BAA^T\right)=\sum_i a_i^T\left(Ba_i\right) \leq a_i^Ta_i=\mathrm{tr}\left(AA^T\right)$$

# 前端里程计方案—基于直接匹配

## ICP系列—点到点ICP

目的：找到$R$，把$\mathrm{Trace}(RH)$ 转换成$\mathrm{Trace}\left(AA^T\right)$ 的形式

方法：

对H进行SVD分解

$$H = U\Sigma V^T$$

取

$$R = VU^T$$

则有

$$RH = VU^TU\Sigma V^T = V\Sigma V^T = V\Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}}V^T = V\Sigma^{\frac{1}{2}}\left(V\Sigma^{\frac{1}{2}}\right)^T$$

旋转$R$确定之后，易得平移为

$$t = u_x - Ru_y$$

注意：我们看公式推导通常是1234按步骤就这么出来了，但是要说思路的话，有时候可能是反推出来的。这个例子就是一个明显的反推

反推：对H和R进行变化，凑成自己想要的形式

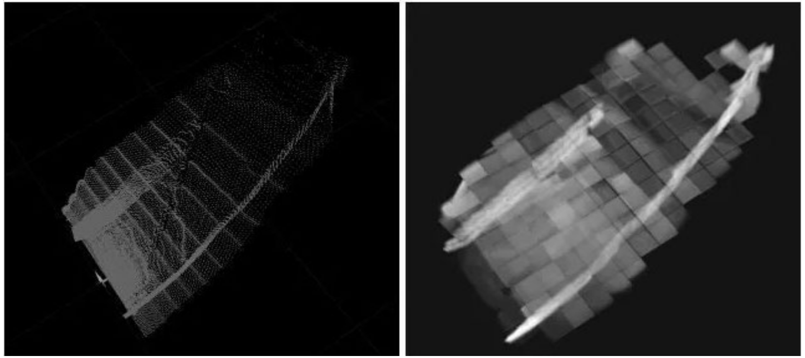
启示：拼凑正定矩阵，可以考虑SVD分解

# 前端里程计方案—基于直接匹配

## ICP系列—其他ICP

# 前端里程计方案—基于直接匹配

## NDT系列—经典NDT



点集：

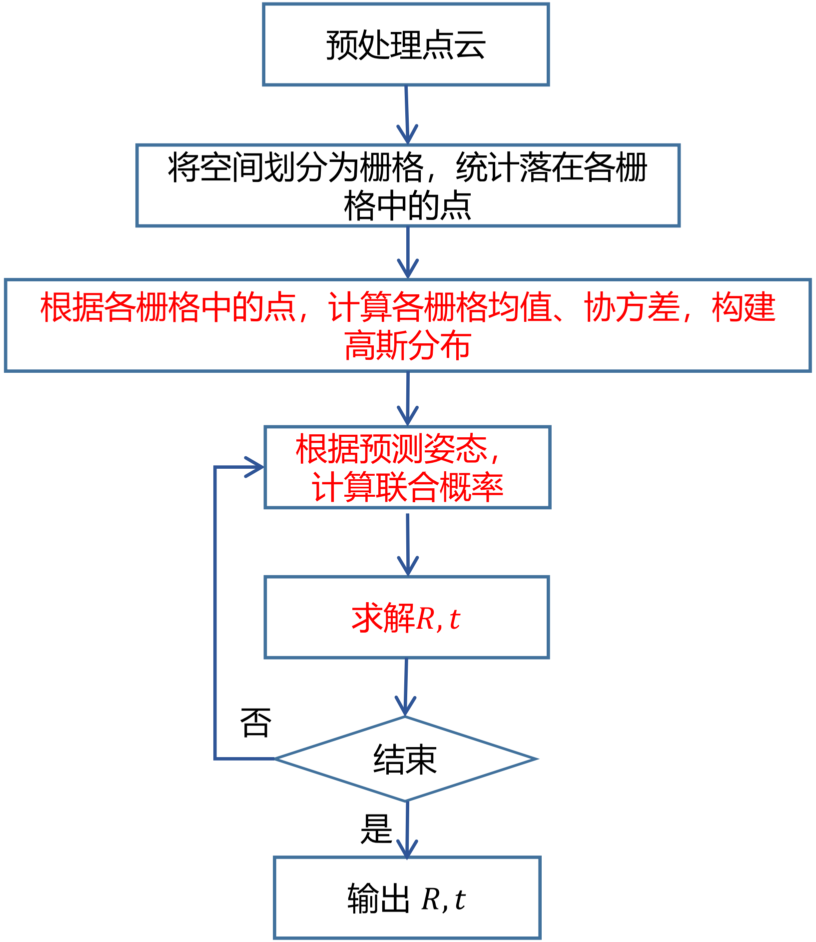
$$X = \{x_1, x_2, \cdots, x_{N_x}\}$$

$$Y = \{y_1, y_2, \cdots, y_{N_y}\}$$

目标： $max\Psi = max \prod_{i=1}^{N_y} f\left(X, T\left(p, y_i\right)\right)$

2D模型： $p = p_3 = [t_x \quad t_y \quad \phi_z]^{\mathrm{T}}$

3D模型： $p = p_6 = [t_x \quad t_y \quad t_z \quad \phi_x \quad \phi_y \quad \phi_z]^{\mathrm{T}}$



# 前端里程计方案—基于直接匹配

## NDT系列—经典NDT

均值:

$$\mu=\frac{1}{N_x}\sum_{i=1}^{N_x}x_i$$

协方差:

$$\mathbf{\Sigma}=\frac{1}{N_x-1}\sum_{i=1}^{N_x}(x_i-\mu)(x_i-\mu)^{\mathrm{T}}$$

根据预测的位姿，对点进行旋转和平移:

$$y_i'=T(p,y_i)=Ry_i+t$$

旋转和平移后的点与目标点集中的点在同一坐标系下，此时可计算各点的概率:

$$f(X,y_i')=\frac{1}{\sqrt{2\pi}\sqrt{|\mathbf{\Sigma}|}}\exp\left(-\frac{(y_i'-\mu)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(y_i'-\mu)}{2}\right)$$

思路：target点云建立高斯分布，source点云往里投

所有点的联合概率:

$$\Psi=\prod_{i=1}^{N_y}f(X,T(p,y_i))$$

$$=\prod_{i=1}^{N_y}\frac{1}{\sqrt{2\pi}\sqrt{|\mathbf{\Sigma}|}}\exp\left(-\frac{(y_i'-\mu)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(y_i'-\mu)}{2}\right)$$

取对数，简化问题:

$$ln\Psi=\sum_{i=1}^{N_y}\left(-\frac{(y_i'-\mu)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(y_i'-\mu)}{2}+\underbrace{\boxed{ln(\frac{1}{\sqrt{2\pi}\sqrt{|\mathbf{\Sigma}|}})}}_{\text{常数}}\right)$$

去除常数项:

$$max\Psi=maxln\Psi=min\Psi_1=min\sum_{i=1}^{N_y}(y_i'-\mu)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(y_i'-\mu)$$

常见套路：第一步，高斯分布的联合概率写成累乘；第二步，去对数，累乘变成指数部分的累加；第三步，解最小二成问题。

# 前端里程计方案—基于直接匹配

## NDT系列—经典NDT

目标函数：$min\sum_{i=1}^{N_y}(y_i'-\mu)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(y_i'-\mu)$

$$y_i'=T\left(p,y_i\right)=Ry_i+t$$

待求参数：$R,t$

令：$e_i(p)=y_i'-\mu$

$$\begin{aligned}F_i(p)&=e_i^{\mathrm{T}}(p)\mathbf{\Sigma}^{-1}e_i(p)\\&=(y_i'-\mu)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(y_i'-\mu)\end{aligned}$$

则：$min\sum_{i=1}^{N_y}(y_i'-\mu)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(y_i'-\mu)$

$$=min\sum_{i=1}^{N_y}F_i(p)$$

迭代优化，即找到 $\Delta p$ 使下式中的值达到最小

$$\sum_{i=1}^{N_y}F_i(p+\Delta p)=\sum_{i=1}^{N_y}e_i^{\mathrm{T}}(p+\Delta p)\mathbf{\Sigma}^{-1}e_i(p+\Delta p)$$

泰勒展开：

$$\begin{aligned}e_i\left(p+\Delta p\right)&\approx e_i\left(p\right)+\underline{\frac{de_i}{dp}}\Delta p\\&=e_i+J_i\Delta p\end{aligned}$$

$$\begin{aligned}F_i\left(p+\Delta p\right)&=e_i\left(p+\Delta p\right)^T\mathbf{\Sigma}^{-1}e_i\left(p+\Delta p\right)\\&\approx\left(e_i+J_i\Delta p\right)^T\mathbf{\Sigma}^{-1}\left(e_i+J_i\Delta p\right)\\&=e_i^T\mathbf{\Sigma}^{-1}e_i+2\underline{e_i^T\mathbf{\Sigma}^{-1}J_i}\Delta p+\Delta p^T\underline{J_i^T\mathbf{\Sigma}^{-1}J_i}\Delta p\\&=F_i(p)+2b_i^T\Delta p+\Delta p^TH_i\Delta p\end{aligned}$$

其中，

$$\begin{aligned}b_i^T&=e_i^T\mathbf{\Sigma}^{-1}J_i\\H_i&=J_i^T\Sigma^{-1}J_i\end{aligned}$$

目标函数随自变量的变化为：

$$\Delta F_i(p)=F_i(p+\Delta p)-F_i(p)=2b_i^T\Delta p+\Delta p^TH_i\Delta p$$

# 前端里程计方案—基于直接匹配

## NDT系列—经典NDT

上述优化问题，转化为，找到 $\Delta p$ 使 $\Delta F_i(p)$ 取得极小值。

令其导数为零：

$$\frac{d\Delta F_i(p)}{d\Delta p} = 2b_i + 2H_i\Delta p = 0$$

即

$$H_i\Delta p = -b_i$$

根据定义：

$$b_i^T = e_i^T \boldsymbol{\Sigma}^{-1} J_i$$

$$H_i = J_i^T \Sigma^{-1} J_i$$

就是高斯牛顿的壳子
壳子里面是雅克比矩阵，这个需
要自己具体来推，完毕

因此，只需得到 $J_i$ ，即可求得 $\Delta p$

$$J_i = \frac{de_i}{dp}$$

# 前端里程计方案—基于直接匹配

## NDT系列—经典NDT

2D场景：

$$p = [t_x \quad t_y \quad \phi_z]^{\mathrm{T}}$$

$$\begin{aligned} y_i' &= T(p, y_i) \\ &= \begin{bmatrix} \cos\phi_z & -\sin\phi_z \\ \sin\phi_z & \cos\phi_z \end{bmatrix} y_i + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \end{aligned}$$

$$e_i = y_i' - \mu$$

雅克比 $J_i = \begin{bmatrix} 1 & 0 & -y_{i1}\sin\phi_z - y_{i2}\cos\phi_z \\ 0 & 1 & y_{i1}\cos\phi_z - y_{i2}\sin\phi_z \end{bmatrix}$

3D场景：

$$p = [t_x \quad t_y \quad t_z \quad \phi_x \quad \phi_y \quad \phi_z]^{\mathrm{T}}$$

$$\begin{aligned} y_i' &= T(p, y_i) \\ &= R_x R_y R_z y_i + t \\ &= \begin{bmatrix} c_y c_z & -c_y s_z & s_y \\ c_x s_z + s_x s_y c_z & c_x c_z - s_x s_y s_z & -s_x c_y \\ s_x s_z - c_x s_y c_z & c_x s_y s_z + s_x c_z & c_x c_y \end{bmatrix} y_i + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \end{aligned}$$

$$e_i = y_i' - \mu$$

雅克比 $J_i = \begin{bmatrix} 1 & 0 & 0 & 0 & c & f \\ 0 & 1 & 0 & a & d & g \\ 0 & 0 & 1 & b & e & h \end{bmatrix}$

其中

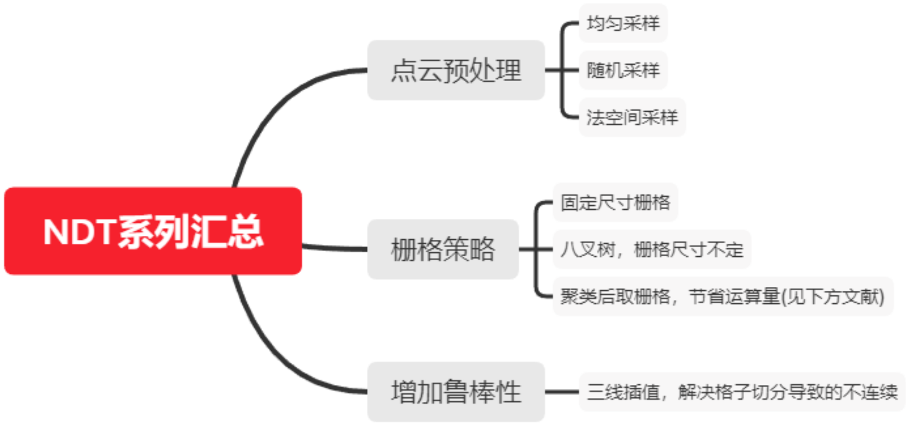
$$a = y_{i1}(-s_x s_z + c_x s_y c_z) + y_{i2}(-s_x c_z - c_x s_y s_z) + y_{i3}(-c_x c_y)$$
$$b = y_{i1}(c_x s_z + s_x s_y c_z) + y_{i2}(-s_x s_y s_z + c_x c_z) + y_{i3}(-s_x c_y)$$
$$c = y_{i1}(-s_y c_z) + y_{i2}(s_y s_z) + y_{i3}(c_y)\,;\, d = y_{i1}(s_x c_y c_z) + y_{i2}(-s_x c_y s_z) + y_{i3}(s_x s_y)$$
$$e = y_{i1}(-c_x c_y c_z) + y_{i2}(c_x c_y s_z) + y_{i3}(-c_x s_y)\,;\, f = y_{i1}(-c_y s_z) + y_{i2}(-c_y c_z)$$
$$g = y_{i1}(c_x c_z - s_x s_y s_z) + y_{i2}(-c_x s_z - s_x s_y c_z)\,;\, h = y_{i1}(s_x c_z + c_x s_y s_z) + y_{i2}(c_x s_y c_z - s_x s_z)$$

# 前端里程计方案—基于直接匹配

## NDT系列—其他NDT

工程上，先考虑能不能换别的方法，再考虑能不能在本方法上深挖。

LOAM方法在起点上，比NDT要高




1. Scan Registration using Segmented Region Growing NDT. Das A, Waslander SL.  2014.

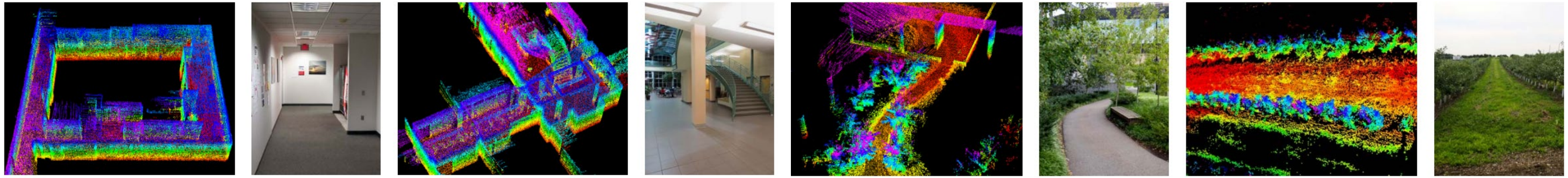
2. 3D Scan Registration Using the Normal Distributions Transform with Ground Segmentation and Point Cloud Clustering. Das A, Waslander SL. 2013.

3. Scan Registration with Multi-Scale K-Means Normal Distributions Transform. Das A, Waslander SL.  2012.

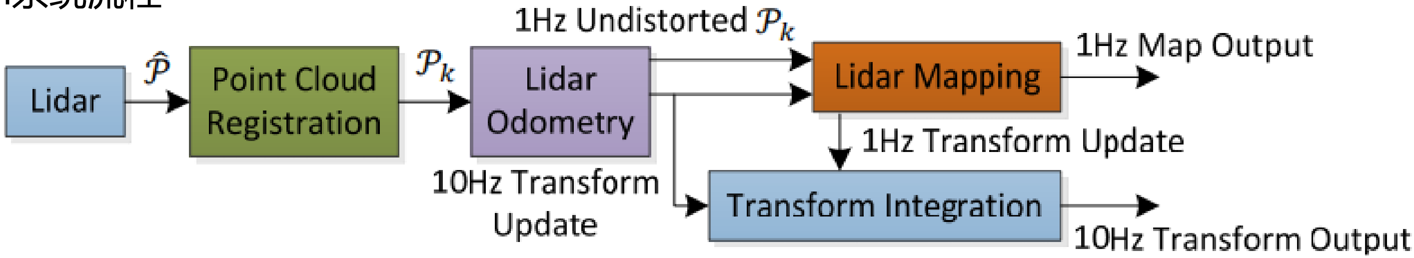
# 前端里程计方案—基于特征

总评：精度和效率都比其他的方法要高

## LOAM系列—LOAM



LOAM系统流程




整体思想：在点云中提取边缘特征(如树干、墙角等，采用线模型描述)和平面特征(如地面、墙面等，采用面模型描述)，匹配时以点到线距离和点到面距离为残差优化位姿。

LOAM: Lidar Odometry and Mapping in Real-time, Ji Zhang and Sanjiv Singh
Code: https://github.com/cuitaixiang/LOAM_NOTED

# 前端里程计方案—基于特征

## LOAM系列—LOAM

1. 提取特征

1) 按线数分割 如果能保留原始的ring信息，则不需要按照xyz来得到ring

**Figure 9-1** VLP-16 Sensor Coordinate System

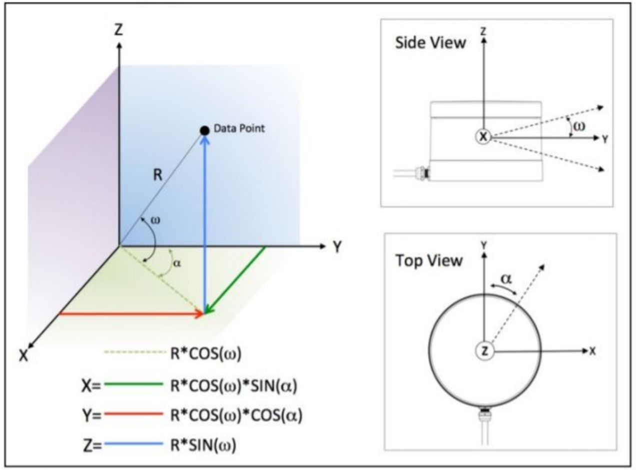
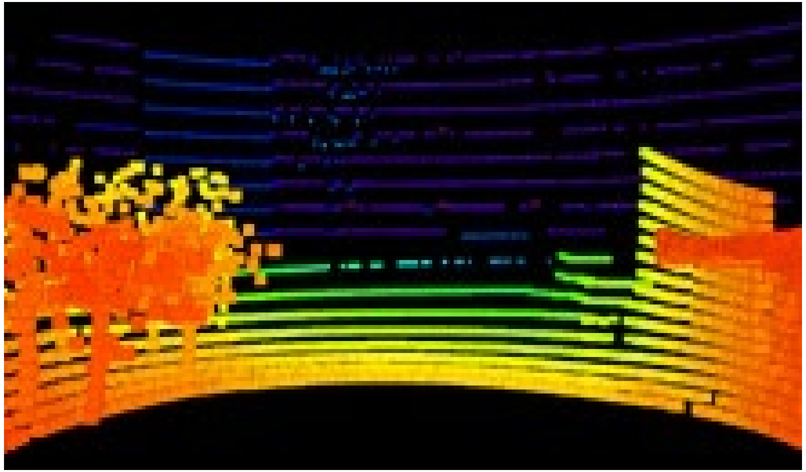



*Table 9-1 below* lists the fixed vertical/elevation angles for each laser in the sensor, along with vertical corrections. The set of angles vary by sensor model (see *Factory Bytes on page 56* for more).



# 前端里程计方案—基于特征

## LOAM系列—LOAM

1. 提取特征

2) 计算曲率

$$c = \frac{1}{|\mathcal{S}| \cdot \left\| \boldsymbol{X}^L_{(k,i)} \right\|} \left\| \sum_{j \in \mathcal{S}, j \neq i} \left( \boldsymbol{X}^L_{(k,i)} - \boldsymbol{X}^L_{(k,j)} \right) \right\|$$

该公式计算点$i$的曲率，其中$S$代表在$i$周围，并且和$i$在同一条扫描线上的点，$\boldsymbol{X}^L_{(k,i)}$ 和 $\boldsymbol{X}^L_{(k,j)}$ 分别代表点$i$和$j$的坐标。

3) 删除异常点

# 前端里程计方案—基于特征

## LOAM系列—LOAM

1. 提取特征

4) 按曲率大小筛选特征点

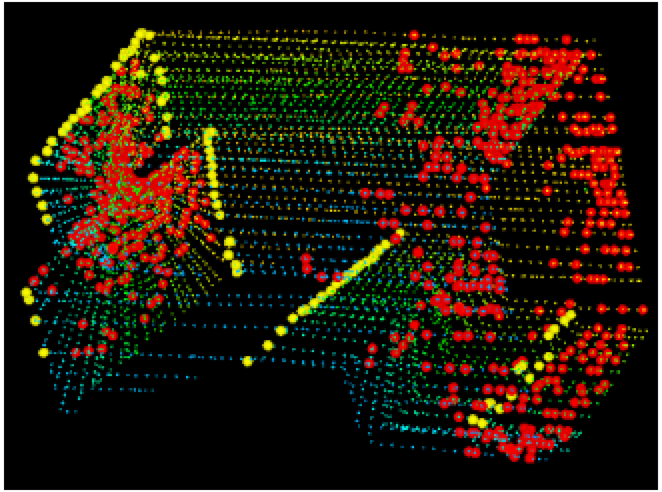
共分4类：

a. 曲率特别大的点(sharp)

b. 曲率大的点(less_shap)

c. 曲率特别小的点(flat)

d. 曲率小的点(less_flat)



典型效果图

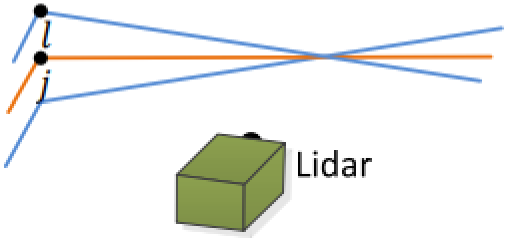
# 前端里程计方案—基于特征

## LOAM系列—LOAM

2. 帧间匹配

1) 特征关联与损失函数计算

a. 线特征

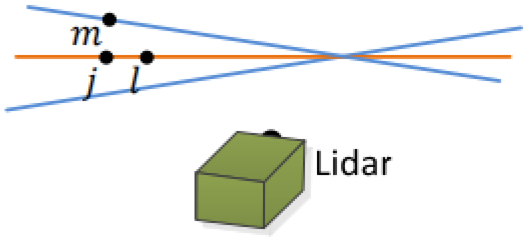



点到线的距离

$$d_{\mathcal{E}} = \frac{\left|\left(\tilde{\boldsymbol{X}}_{(k+1,i)}^{L} - \overline{\boldsymbol{X}}_{(k,j)}^{L}\right) \times \left(\tilde{\boldsymbol{X}}_{(k+1,i)}^{L} - \overline{\boldsymbol{X}}_{(k,l)}^{L}\right)\right|}{\left|\overline{\boldsymbol{X}}_{(k,j)}^{L} - \overline{\boldsymbol{X}}_{(k,l)}^{L}\right|} \quad (1)$$

很好理解：
分子是一个叉乘，其模是平行四边形的面积
其余迎刃而解

b. 面特征




点到面的距离

体积除以底面积

$$d_{\mathcal{H}} = \frac{\left|\begin{array}{c}(\tilde{\boldsymbol{X}}_{(k+1,i)}^{L} - \bar{\boldsymbol{X}}_{(k,j)}^{L}) \\ ((\bar{\boldsymbol{X}}_{(k,j)}^{L} - \bar{\boldsymbol{X}}_{(k,l)}^{L}) \times (\bar{\boldsymbol{X}}_{(k,j)}^{L} - \bar{\boldsymbol{X}}_{(k,m)}^{L}))\end{array}\right|}{\left|(\bar{\boldsymbol{X}}_{(k,j)}^{L} - \bar{\boldsymbol{X}}_{(k,l)}^{L}) \times (\bar{\boldsymbol{X}}_{(k,j)}^{L} - \bar{\boldsymbol{X}}_{(k,m)}^{L})\right|} \quad (2)$$

把(1)和(2)放在一个模型里，得到总的损失函数为

$$loss = \sum_{i=1}^{N_{\mathcal{E}}} d_{\mathcal{E}i} + \sum_{i=1}^{N_{\mathcal{H}}} d_{\mathcal{H}i} = D\left(\tilde{X}_{k+1,i}^{L}\right) (3)$$

# 前端里程计方案—基于特征

## LOAM系列—LOAM

### 2. 帧间匹配

### 1) 特征关联与损失函数计算

定义 $t_{k+1}$ 时刻的位姿为

$$T_{k+1}^{L}=[t_x,t_y,t_z,\theta_x,\theta_y,\theta_z]^T$$

特征点从当前雷达坐标系投影到目标坐标系

$$\begin{aligned}\tilde{\boldsymbol{X}}_{(k+1,i)}^{L}&=\mathbf{R}\boldsymbol{X}_{(k+1,i)}^{L}+t\\&=G\left(\boldsymbol{X}_{(k+1,i)}^{L},\boldsymbol{T}_{k+1}^{L}\right)\end{aligned}\quad(4)$$

其中

$$t=[t_x,t_y,t_z]^T$$

$$\begin{aligned}R&=R_yR_xR_z\\&=\begin{bmatrix}c_yc_z+s_ys_xs_z & c_zs_ys_x-c_ys_z & c_xs_y\\ c_xs_z & c_xc_z & -s_x\\ c_ys_xs_z-c_zs_y & c_yc_zs_x+s_ys_z & c_yc_x\end{bmatrix}\end{aligned}$$

又

$$\begin{cases}c_x=cos(\theta_x)\\ s_x=sin(\theta_x)\\ c_y=cos(\theta_y)\\ s_y=sin(\theta_y)\\ c_z=cos(\theta_z)\\ s_z=sin(\theta_z)\end{cases}$$

合并(3)和(4)得到

$$loss=F\left(\boldsymbol{X}_{(k+1,i)}^{L},\boldsymbol{T}_{k+1}^{L}\right)=D\left(G\left(\boldsymbol{X}_{(k+1,i)}^{L},\boldsymbol{T}_{k+1}^{L}\right)\right)(5)$$

# 前端里程计方案—基于特征

## LOAM系列—LOAM

2. 帧间匹配

2) LM迭代优化

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

其中

$$\begin{aligned}
\mathbf{J} &= \frac{\partial \boldsymbol{F}\left(\boldsymbol{X}_{(k+1,i)}^{L}, \boldsymbol{T}_{k+1}^{L}\right)}{\partial \boldsymbol{T}_{k+1}^{L}} \\
&= \frac{\partial D\left(G\left(\boldsymbol{X}_{(k+1,i)}^{L}, \boldsymbol{T}_{k+1}^{L}\right)\right)}{\partial \boldsymbol{T}_{k+1}^{L}} \\
&= \frac{\partial D(\tilde{X}_{k+1,i}^{L})}{\partial \tilde{\boldsymbol{X}}_{(k+1,i)}^{L}} \frac{\partial G\left(\boldsymbol{X}_{(k+1,i)}^{L}, \boldsymbol{T}_{k+1}^{L}\right)}{\partial \boldsymbol{T}_{k+1}^{L}}
\end{aligned}$$

对于线特征, 梯度方向为通过特征点的垂直于直线的方向

$$\frac{\partial D(\tilde{X}_{k+1,i}^{L})}{\partial \tilde{\boldsymbol{X}}_{(k+1,i)}^{L}} = [a_{\mathcal{E}}, b_{\mathcal{E}}, c_{\mathcal{E}}]^{T}$$

对于面特征, 梯度方向为通过特征点的垂直于平面的方向

$$\frac{\partial D(\tilde{X}_{k+1,i}^{L})}{\partial \tilde{\boldsymbol{X}}_{(k+1,i)}^{L}} = [a_{\mathcal{H}}, b_{\mathcal{H}}, c_{\mathcal{H}}]^{T}$$

# 前端里程计方案—基于特征

## LOAM系列—LOAM

2. 帧间匹配

2) LM迭代优化

对平移求导

$$
\begin{aligned}
& \frac{\partial G\left(\boldsymbol{X}_{(k+1,i)}^{L}, \boldsymbol{T}_{k+1}^{L}\right)}{\partial \boldsymbol{t}_x} \\
= & \frac{\partial G\left(\mathbf{R}\boldsymbol{X}_{(k+1,i)}^{L}+t\right)}{\partial \boldsymbol{t}_x} \\
= & \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}
\end{aligned}
$$

同理，可推导另外两个方向位移的导数

对角度求导

$$
\begin{aligned}
& \frac{\partial G\left(\boldsymbol{X}_{(k+1,i)}^{L}, \boldsymbol{T}_{k+1}^{L}\right)}{\partial \boldsymbol{\theta}_x} \\
= & \frac{\partial G\left(\mathbf{R}\boldsymbol{X}_{(k+1,i)}^{L}+t\right)}{\partial \boldsymbol{\theta}_x} \\
= & \frac{\partial G\left(\mathbf{R}\boldsymbol{X}_{(k+1,i)}^{L}\right)}{\partial \boldsymbol{\theta}_x} \\
= & \begin{bmatrix} s_y c_x s_z & c_z s_y c_x & -s_x s_y \\ -s_x s_z & -s_x c_z & -c_x \\ c_y c_x s_z & c_y c_z c_x & -c_y s_x \end{bmatrix} \begin{bmatrix} x_{(k+1,i)} \\ y_{(k+1,i)} \\ z_{(k+1,i)} \end{bmatrix}
\end{aligned}
$$

同理，可推导另外两个角度的导数

# 前端里程计方案—基于特征

## LOAM系列—LOAM

3. 构建地图

1) 合并地图点

a. 把关键帧的特征点按照位姿转到地图坐标系中

b. 按照位置和cube尺寸划分到对应的cube中

2) 位姿优化与里程计的方法同理

代码：https://github.com/cuitaixiang/LOAM_NOTED

## LOAM系列—ALOAM

主要特点

1) 去掉了和IMU相关的部分
2) 使用Eigen做位姿转换，简化了代码
3) 使用ceres做迭代优化，简化了代码，但降低了效率

代码：https://github.com/HKUST-Aerial-Robotics/A-LOAM

## LOAM系列—FLOAM

主要特点

1) 整体和ALOAM类似，只是把ceres中的自动求导，转变成了直接使用推导的雅克比矩阵

代码：https://github.com/wh200720041/floam

# 前端里程计方案—基于特征

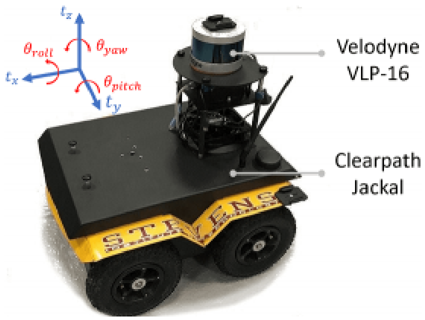
## LOAM系列—LEGO-LOAM

1. 主要特点

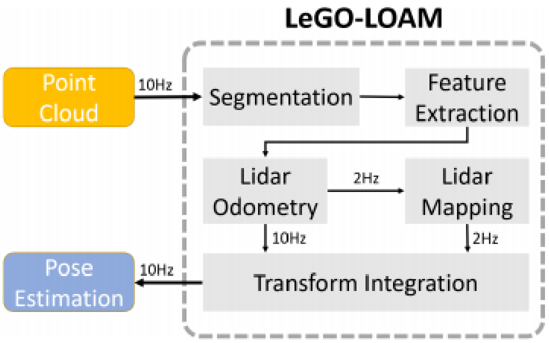
1) 对地面做了分割，减小了特征搜索范围

2) 提取特征之前做了聚类，提高了特征质量

3) 水平和航向分别优化，提高了效率




(a) Jackal UGV




(b) System overview

# 前端里程计方案—基于特征

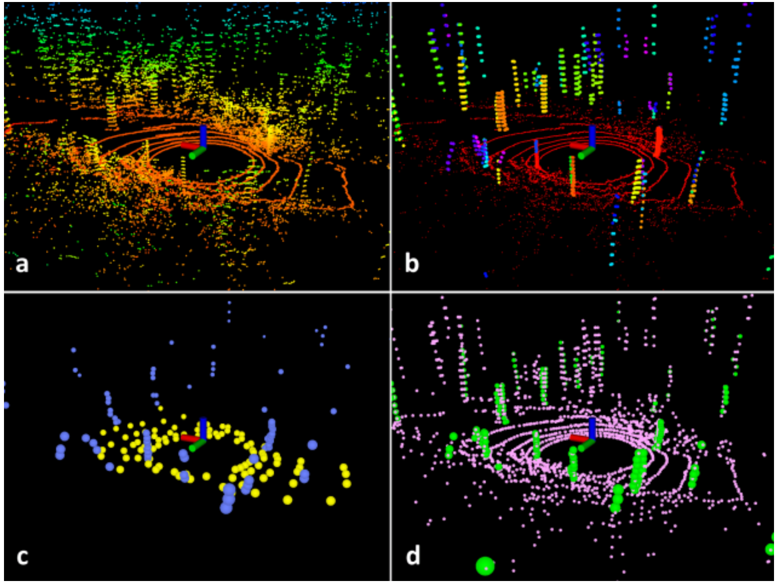
## LOAM系列—LEGO-LOAM

2. 特征提取

1) 根据线与线之间的夹角，以及点的曲率，筛选出地面点。所有用于匹配的平面点仅使用地面点。

2) 在非地面点中，使用广度优先搜索(BFS)做聚类，聚类中点的数量大于30，才用来筛选线特征。

3) 筛选线特征方法，与LOAM中相同



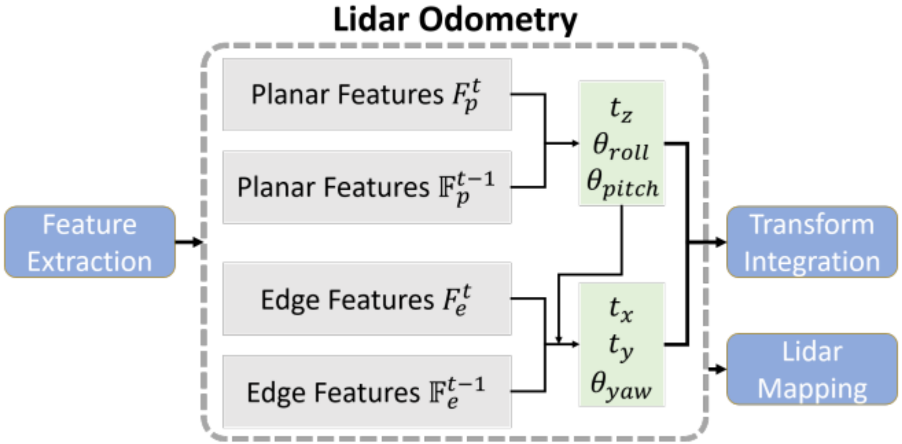
# 前端里程计方案—基于特征

## LOAM系列—LEGO-LOAM

3. 里程计

1) 使用地面面特征优化高度和水平角

2) 使用线特征优化水平位移和航向角




具体推导过程，可根据LOAM中的六自由度方法，重新推导此处三自由度方法

代码：https://github.com/RobustFieldAutonomyLab/LeGO-LOAM

# 目录

# 基于数据集实现

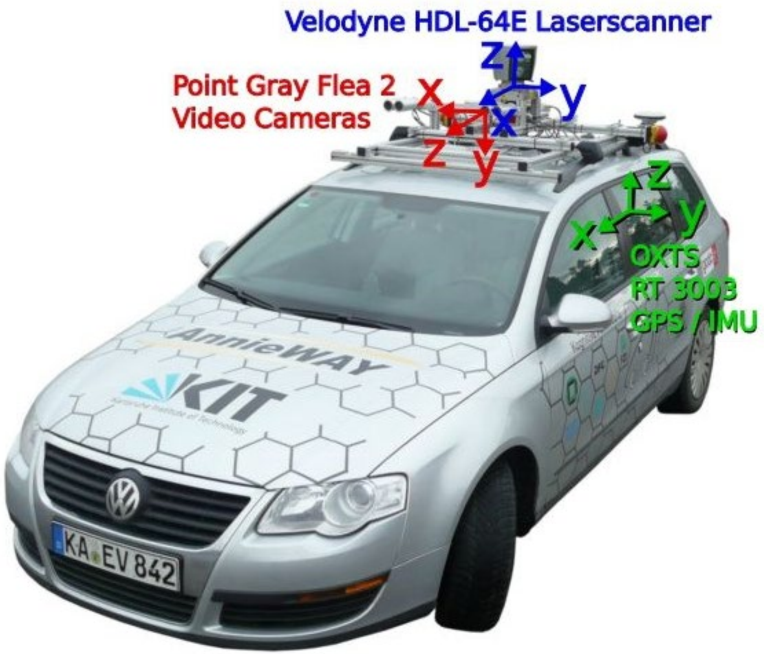
## 1. KITTI数据集简介

硬件组成：

1) 一个64线激光雷达，在车顶的正中心

2) 两个彩色摄像头和两个黑白摄像头，在雷达两侧。

3) 一个组合导航系统（OXTS RT 3003），在雷达左后方。它可以输出RTK/IMU组合导航结果，包括经纬度和姿态，同时也输出IMU原始数据。

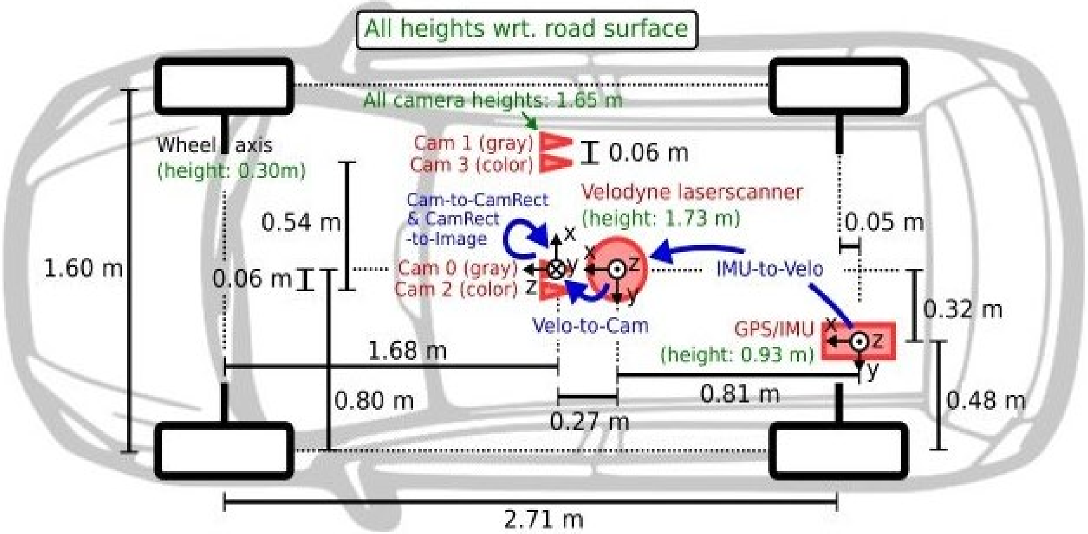


# 基于数据集实现

## 1. KITTI数据集简介

安装关系：

图中所示的所有安装关系，都可以在数据集提供标定文件中找到，可直接使用。



# 基于数据集实现

## 2. 数据集使用

本课程以ubuntu16.04下的ros kinetic为调试环境，因此需要把数据制作成ros的bag文件。

1) 升级numpy

kitti2bag要求numpy版本>=1.12，ubuntu 16.04

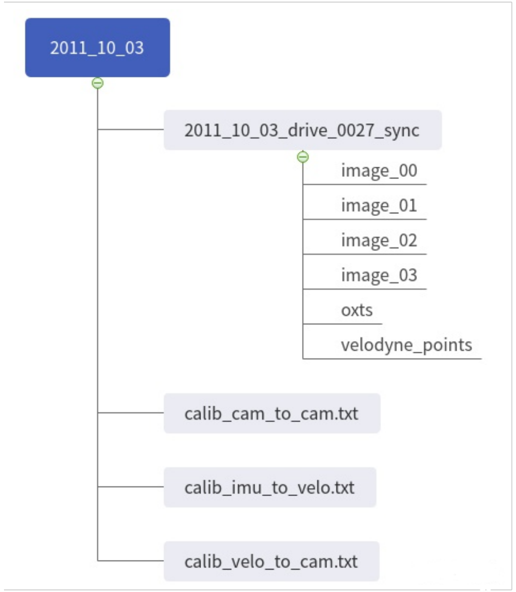
默认的是1.11，升级可以通过一条指令来完成

```
sudo pip install -U numpy
```

2) 安装kitti2bag

```
sudo pip install kitti2bag
```

3) 按以下目录存放文件

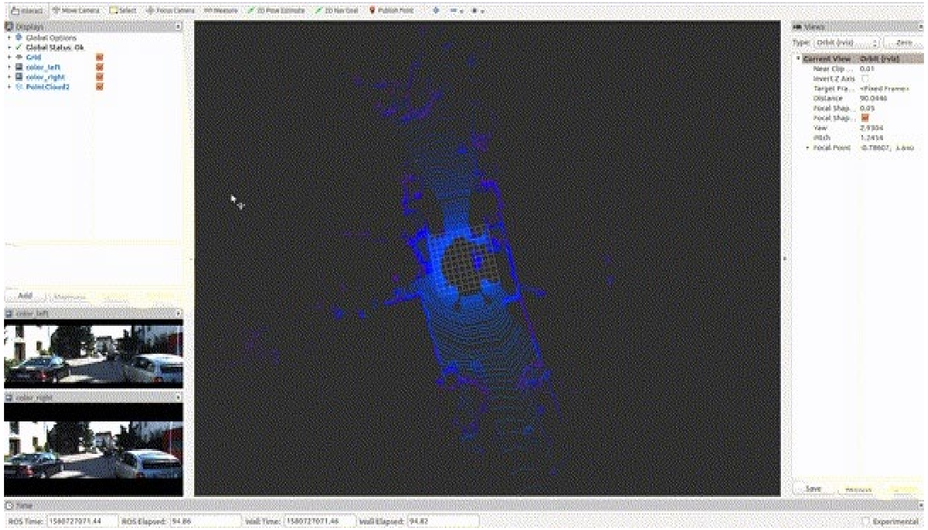


# 基于数据集实现

## 2. 数据集使用

4) 生成bag

kitti2bag -t 2011_10_03 -r 0027 raw_synced



5) 测试bag

a. roscore
b. rviz
c. rosbag play
kitti_2011_10_03_drive_0027_synced.bag

参考文章：[从零开始做自动驾驶定位(二): 数据集]

# 基于数据集实现

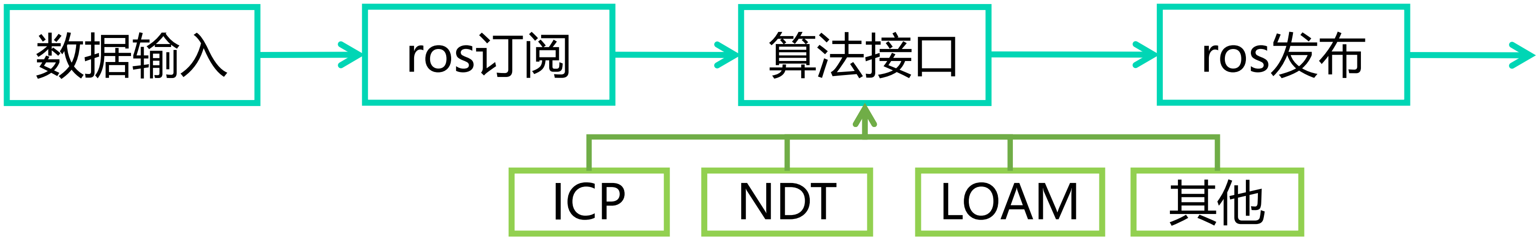
## 3. 里程计工程框架实现

核心思想：

1) 通过类的封装，实现模块化

2) 把ros流程与c++内部实现分开，使流程清晰

3) 基于c++多态，实现高可扩展性。




参考文章：

[从零开始做自动驾驶定位(三): 软件框架]

[从零开始做自动驾驶定位(四): 前端里程计之初试]

[从零开始做自动驾驶定位(五): 前端里程计之代码优化]

# 基于数据集实现

## 4. 里程计精度评价

以组合导航的结果为真值，使用evo工具进行里程计精度评价

### 1) 安装evo

```
pip install evo --upgrade --no-binary evo
```

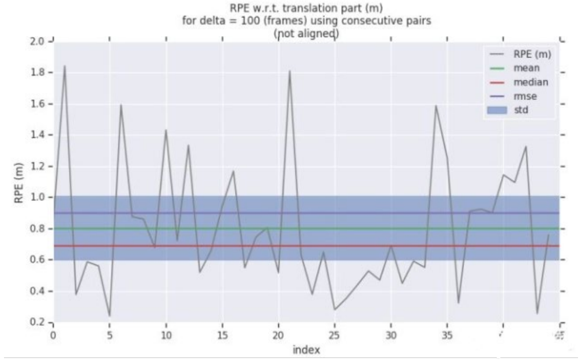
### 2) 使用evo计算轨迹误差

a. 分段统计精度

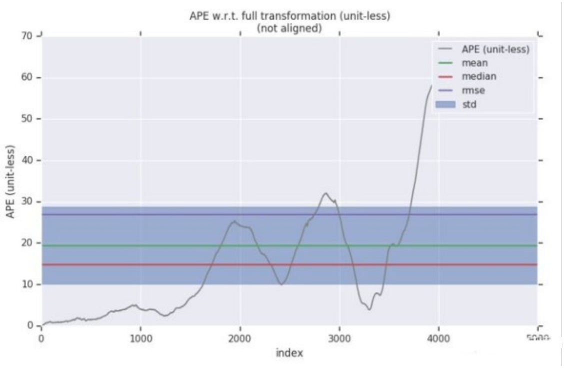
```
evo_rpe kitti ground_truth.txt laser_odom.txt -r trans_part --delta 100 --plot --plot_mode xyz
```

b. 计算整体轨迹误差

```
evo_ape kitti ground_truth.txt laser_odom.txt -r full --plot --plot_mode xyz
```




分段统计精度




整体轨迹误差

深蓝学院
shenlanxueyuan.com

# 感谢聆听！

Thanks for Listening