Jindong_Shi 2022.04.10

对多种 ICP 点云匹配方法 (ICP, PL-ICP, NICP, IMLS-ICP) 的比较

ICP (Iterative Closest Point) 迭代最近点

参考文献[1]

作为 ICP 类点云匹配方法的始祖，ICP 方法是一种最通用的求解两个点云集合转换关系的方法，实现的是点到点的点云匹配。

倘若给定两个已经匹配好的点云集合: 参考帧(或目标帧)内的点集 $X = \{x_i\}$，$i = 1, 2, \ldots, N$，当前帧内的点集 $P = \{p_i\}$，$i = 1, 2, \ldots, N$

这里 x_i 和 p_i 可以是 2x1 或 3x1 的列向量，且假设两个点集之间已经做好匹配，即 x_1 对应 p_1, x_2 对应 p_2, ..., x_N 对应 p_N

我们希望找到最佳的旋转矩阵 R 和平移向量 t，使得将当前帧点集 P 旋转到目标点集 X 的考虑欧式距离的如下目标损失函数到达最小:

$$E(R,t) = \frac{1}{N_p}\sum_{i=1}^{N_p}\|x_i - Rp_i - t\|^2$$

我们可以使用基于 SVD 分解的方法按照如下步骤求得闭式解:

Step1:

计算两个点集的几何中心:

$$u_x = \frac{1}{N_p}\sum_{i=1}^{N_p} x_i \qquad u_p = \frac{1}{N_p}\sum_{i=1}^{N_p} p_i$$

将两个点集去中心化:

$$x_i' = x_i - u_x$$
$$p_i' = p_i - u_p$$

Step2:

转化目标损失函数:

$$\begin{aligned}
E(R,t) &= \frac{1}{N_p}\sum_{i=1}^{N_p}\|x_i - Rp_i - t\|^2 \\
&= \frac{1}{N_p}\sum_{i=1}^{N_p}\|x_i - Rp_i - t - u_x + Ru_p + u_x - Ru_p\|^2 \\
&= \frac{1}{N_p}\sum_{i=1}^{N_p}\|x_i - u_x - R(p_i - u_p) + (u_x - Ru_p - t)\|^2 \\
&= \frac{1}{N_p}\sum_{i=1}^{N_p}\left[\|x_i - u_x - R(p_i - u_p)\|^2 + \|u_x - Ru_p - t\|^2 + 2\left(x_i - u_x - R(p_i - u_p)\right)^T (u_x - Ru_p - t)\right] \\
&= \frac{1}{N_p}\sum_{i=1}^{N_p}\|x_i - u_x - R(p_i - u_p)\|^2 + \|u_x - Ru_p - t\|^2 + \frac{1}{N_p}\sum_{i=1}^{N_p} 2\left(x_i - u_x - R(p_i - u_p)\right)^T (u_x - Ru_p - t)
\end{aligned}$$

红色框的内容等于 0，所以目标函数只与两个蓝色框内的内容有关。

在第二个蓝色框内，对于任意一个 R，总能找到 $t = u_x - Ru_p$, 使得该蓝色框取到最小值 0。

而第一个蓝色框的大小只与 R 有关。综上，若要使得损失函数 E(R,t)整体最小，只需要找到使第一个蓝色框最小的 R 既可，后续 t 可以根据 $t = u_x - Ru_p$ 求得。

所以我们关注第一个蓝色框:

$$minE_1(R,t) = \frac{1}{N_p}\sum_{i=1}^{N_p}\left\|x_i - u_x - R(p_i - u_p)\right\|^2$$
$$= \frac{1}{N_p}\sum_{i=1}^{N_p}\|x_i' - Rp_i'\|^2$$
$$= \frac{1}{N_p}\sum_{i=1}^{N_p} x_i'^T x_i' + p_i'^T R^T R p_i' - 2x_i'^T R p_i'$$
$$= \frac{1}{N_p}\sum_{i=1}^{N_p} x_i'^T x_i' + p_i'^T p_i' - 2x_i'^T R p_i'$$
$$= \frac{1}{N_p}\sum_{i=1}^{N_p} -2x_i'^T R p_i' + \underbrace{\frac{1}{N_p}\sum_{i=1}^{N_p}(x_i'^T x_i' + p_i'^T p_i')}_{\text{与}R\text{无关}}$$

由于上述第二项与 R 无关，所以最小化 $E_1(R,t)$相当于最大化:

$$max\sum_{i=1}^{N_p} x_i'^T R p_i'$$

$\sum_{i=1}^{N_p} x_i'^T R p_i'$ 在这里是标量，而对于标量我们可以在前面加上 Trace, 所以有:

$$max\sum_{i=1}^{N_p} x_i'^T R p_i' = max\sum_{i=1}^{N_p}\mathrm{Trace}(Rp_i'x_i'^T) = max\mathrm{Trace}(R(\underbrace{\sum_{i=1}^{N_p} p_i'x_i'^T}_{\text{令其为}H}))$$
$$= maxTrace(RH)$$

至此最小化 E(R,t)的问题转化为了最大化 RH 的迹的问题。那么如何求得 RH 的迹的最大值呢?这里我们需要引入一个定理:

对于任意一个正定对称矩阵 AA^T和任意正交矩阵 B，均满足:

$$\mathrm{Trace}(AA^T) \geq \mathrm{Trace}(BAA^T)$$

证明该定理:

令 ai 为矩阵 A 的第 i 个列向量，那么

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

根据 Schwarz 不等式:

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

所以:

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

根据该引用定理可知，假如我们能够找一个正交旋转矩阵 R，使得 RH 能转化为 AA^T的形式，那么在此基础上再乘上其他任意一个旋转矩阵(正交矩阵)，都不会使得 $\mathrm{Trace}(AA^T)$变得更大了。换言之，能使 RH 转化为 AA^T形式的 R 就是我们要找的使得 Trace(RH)最大的旋转矩阵!

Step3:

计算 H 的 SVD 分解，即 $H = U\Lambda V^T$

构建正交矩阵 X，令 $X = VU^T$

因为 $XH = VU^TU\Lambda V^T = V\Lambda V^T = V\Lambda^{1/2}(V\Lambda^{1/2})^T$ 转化为了 AA^T的形式

所以:

当 det(X) = +1 时, X 满足作为一个旋转矩阵的要求。

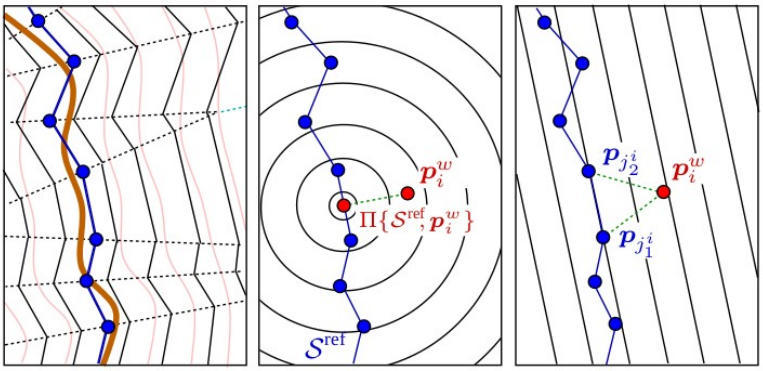
当 det(X) = -1 时, X 包含一个反射而不是真旋转矩阵。(这种情况几乎不会出现)

综上，当 det(X) = +1, 且 $R = X = VU^T$ 时，损失函数 E(R,t)最小。
我们最终得到两个点云集合的最小二乘匹配解为:

$$R = VU^T$$
$$t = u_x - Ru_p$$

通常情况下，我们无法得到两个已经匹配好的点云，因而无法一步求得闭式解，我们只能通过迭代的方法不断去接近最优解，即一边匹配，一边计算位姿。算法流程如下:
1. 寻找当前帧的点集 P 经估计的旋转矩阵 R 和平移向量 t 变换后的点集$\{p_i'\}$，和目标点集 X 中离的最近的点。一般可以使用 KD 树实现快速的最近点搜寻。
2. 根据该匹配关系，计算 R 和 t，既可以使用上述的闭式解，也可以使用如高斯牛顿，LM 等方法求解
3. 计算对点云 P 进行姿态 R 和 t 变换后的损失函数 E(R,t)
4. 假如损失函数小于阈值或迭代步骤大于一个阈值，就结束迭代; 否则，回到步骤 1，继续进行迭代

PL-ICP (Point-to-Line ICP) 点线迭代最近点法
参考文献[2]
ICP 方法实现的是点到点的匹配，但是激光点是对真实环境中曲面的离散采样，而由于前后两帧点云采样的差异，导致点到点的距离有时是不合理的。比如下图中给定曲面 S^{ref} 和点 p_i^w，假如我们采用点到点的匹配方式，点 p_i^w 会匹配到离它最近的两个点 p_{j1}^i 和 p_{j2}^i，而这个点到点的距离由于采样的随机性，往往是随机变化的，不能真实地反映两帧点云之间的相对距离关系。相对比的，我们在进行匹配的时候，如果采用点 p_i^w 到曲面 S^{ref} 的距离(对于二维点云而言，就是点 p_i^w 到曲线的距离)，那么无论我们采到曲线上的哪个点，这个点线距离都是不变的，换言之点到线的距离更具有代表性，不受采样差异性影响，更为鲁棒。



(a) Distance to curve and to polyline (b) Point-to-point metric (c) Point-to-line metric

所以现在的问题在于如何恢复曲面 S^{ref}。PL-ICP 的思想是，用分段线性的方法对真实曲面 S^{ref} 进行近似，即用激光点到目标激光点云中最近的两点连线的距离来模拟实际激光点到曲面的距离。比如上图(c)中，点 p_i^w 到曲面 S^{ref} 的距离，可以用点 p_i^w 到线段 $p_{j1}^i p_{j2}^i$ 的距离来近似。
根据该思想，PL-ICP 的目标函数描述的是多个点到直线的距离的和:

$$\min_{T_k} \sum_i \left\| n_i^T (x_{j2}^i - T_k p_i) \right\|^2 \quad \Leftrightarrow \quad \min_{T_k} \sum_i \left\| x_j^i - T_k p_i \right\|^2$$

其中，$T_k = \begin{bmatrix} R_k & t_k \\ 0 & 1 \end{bmatrix}$ $\quad T_k p_i = R_k p_i + t_k$

该目标函数中:

点 p_i 是当前帧点云集合 P 中的任意一个点，

x_{j1}^i 和 x_{j2}^i 是参考帧点云集合 X 中，距离 p_i 最近的两个点，

n_i^T: 点 x_{j1}^i 和 x_{j2}^i 组成的直线的法向量。

x_j^i: 点 $T_k p_i$ 在直线 $x_{j1}^i x_{j2}^i$ 上的投影

PL-ICP 算法流程如下:

1. 寻找点集 P 中任意一点 p_i，经第 k 次估计的变换矩阵 T_k 变换后的点 $T_k p_i$，和目标点集 X 中离的最近的两个点 x_{j1}^i 和 x_{j2} (一般可以使用 KD 树来实现快速的最近点搜寻)
2. 计算点 $T_k p_i$ 与投影点 x_j^i 的距离，去除误差过大的点
3. 使用 LM 等优化方法，最小化损失函数

$$\min_{T_k} \sum_i \left\| x_j^i - T_k p_i \right\|^2$$

直到满足迭代结束条件，退出迭代。

总结 PL-ICP 相比于 ICP 的优缺点如下:

优点:

a. 在结构化的环境中，PL-ICP 使用点到面的距离更符合实际的情况，求解精度更高。

b. PL-ICP 相比于 ICP 迭代收敛速度更快，因为 PL-ICP 是二阶收敛，而 ICP 是一阶收敛:

$$\left\| T_k - T_\infty \right\| < k \left\| T_{k-1} - T_\infty \right\| \qquad \left\| T_k - T_\infty \right\|^2 < k \left\| T_{k-1} - T_\infty \right\|^2$$

缺点:

c. PL-ICP 对初始值敏感，因此不单独使用，通常与里程计, CSM 等一起使用。一般会将一个不容易陷入局部极值的方法和一个对初值敏感但精度高的方法结合起来，比如 state of art 的做法是先用 CSM 进行暴力搜索得到一个不那么准的估计值，再使用 PL-ICP 方法在该估计值附近计算更准的值。两者互补，效果更加。

NICP (Normal based ICP) 基于法向量的迭代最近点法

参考文献[3]

NICP 在定义误差的时候，除了加入点与点之间的欧式距离之外，还加入了匹配点的法向量之间的距离。换言之，NICP 要求相匹配的两个点不仅要离得足够近，而且它们的方向应该尽可能一致(因为法向量可以代表一个点的方向)，不然就认为它们不是一组好的匹配点。

所以相比于 ICP 只关注平移误差，NICP 还额外关注了角度(即旋转)误差。一般对于一个激光里程计而言，位移的误差对于整体位姿带来的影响是比较小的，但是角度误差对位姿的影响却是毁灭性的。想象一下，一辆小车多驶出或少驶出个几米对小车的位置影响并不很大，但是假如小车的行驶方向偏离个几度，不 y 一会儿之后就会使得小车的位置产生巨大的偏离! 因此，NICP 考虑法向量误差，是对 ICP 方法点云匹配精度的一大改进。

a) 目标损失函数定义

根据上述方法，NICP 定义了有别于 ICP 的目标损失函数。首先 NICP 定义了拓展点。对于某一点 p_i 而言，可以定义它的拓展点，该拓展点不仅包括点 p_i, 还包含该点的方向量 n_i:

$$\tilde{p}_i = \begin{pmatrix} p_i \\ n_i \end{pmatrix}$$

对于该拓展点，还定义了任意变换矩阵 T 对其进行的操作:

$$T \oplus \tilde{p}_i = \begin{pmatrix} Rp_i + t \\ Rn_i \end{pmatrix}$$

好了，那么接下来，对于当前帧的任意一点 p_j 到参考帧内的点 p_i 的误差项可以定义如下, 它包含了匹配点之间的欧式距离和法向量之间的距离。这里论文中给出的公式稍有不同，它是将参考帧内的点 p_j 经变换矩阵 T 变换后，与当前帧的点 p_i 进行误差计算:

$$\mathbf{e}_{ij}(\mathbf{T}) = \left(\tilde{\mathbf{p}}_i^{\mathrm{c}} - \mathbf{T} \oplus \tilde{\mathbf{p}}_j^{\mathrm{r}}\right)$$

至此，目标损失函数可以定义如下，包含了所有匹配点对的误差项:

$$\sum_{\mathcal{C}} \mathbf{e}_{ij}(\mathbf{T})^T \tilde{\mathbf{\Omega}}_{ij} \mathbf{e}_{ij}(\mathbf{T})$$

$$\tilde{\mathbf{\Omega}}_{ij} = \begin{pmatrix} \mathbf{\Omega}_i^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Omega}_i^{\mathrm{n}} \end{pmatrix}$$

$\tilde{\Omega}_{ij}$ 为信息矩阵，对角元素包含了该误差项中欧式距离和法向量距离在优化过程中各自的权重 Ω_i^s 和 Ω_i^n。简单而言，对于那些目标点云中平整的点(即曲率足够小，法向量具有良好的定义的点)，我们应该视其提供了良好的角度信息，因此可以在误差函数中增大其法向量距离的权重 Ω_i^n，来更好地估计角度。假如匹配的目标点云中的某个点是杂乱点(即曲率过大，法向量没有良好的定义的点)，那么我们认为该点的角度信息不应该采信，我们减小 Ω_i^n，而只优化位置信息。

具体而言，信息矩阵的定义是如下:

对于目标帧点云中任意一点 p_i，我们首先找到该点周围半径 R 球形空间内的所有点的点集 V_i (可以使用 kd 树实现更快的搜索)，从而计算该点的均值和协方差，具体定义如下:

$$\mu_i^s = \frac{1}{|\mathcal{V}_i|} \sum_{\mathbf{p}_j \in \mathcal{V}_i} \mathbf{p}_i$$

$$\mathbf{\Sigma}_i^s = \frac{1}{|\mathcal{V}_i|} \sum_{\mathbf{p}_j \in \mathcal{V}_i} (\mathbf{p}_i - \mu_i)(\mathbf{p}_i - \mu_i)^T$$

这里$|V_i|$代表点集 V_i 内点的个数。

我们可以对协方差矩阵做特征值分解，并且将特征值按照从小到大进行排列，即 $\lambda 1 \le \lambda 2 \le \lambda 3$:

$$\mathbf{\Sigma}_i^{\mathrm{s}} = \mathbf{R} \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} \mathbf{R}^T$$

这里列出的是三维点云的协方差，如果是二维点云则只有两个特征值。

紧接着我们可以定义该点的曲率如下:

$$\sigma_i = \lambda_1 / (\lambda_1 + \lambda_2 + \lambda_3)$$

可以想象一下，对于一根很细的长棍而言，它其中一个特征值特别的小，根据上述公式计算得到的曲率会接近于 0，这与我们的经验是相吻合的，因为直线的曲率就是 0。

做完该点 p_i 处的特征分解后，该点的法向量也就可以近似出来了。论文中将该点 p_i 处的法向量近似为最小特征值 λ1 对应的特征向量。如果仔细想一下，这是很有道理的。
至此，我们可以写出点 p_i 处的信息矩阵了：

$$\tilde{\mathbf{\Omega}}_{ij} = \begin{pmatrix} \mathbf{\Omega}_i^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Omega}_i^{\mathrm{n}} \end{pmatrix}$$

$$\Omega_i^s = \left(\sum_i^s\right)^{-1}$$

如果该点曲率足够小，则：

$$\Omega_i^{\mathrm{n}} = \mathbf{R} \begin{pmatrix} \frac{1}{\epsilon} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{R}^T$$

否则 Ω_i^{n} 等于单位矩阵。这里参数 ε 可以调节权重的大小。

b) 点匹配规则：
为了优化角度，NICP 除了把法向量距离考虑进误差项以外，还设计了一系列点匹配规则，使得在迭代过程中可以提前排除一些明显是错误的匹配，来提高优化精度和效率。具体匹配规则如下：

- 如果没有well define的法向量，则拒绝。
- 两点间的距离大于阈值，则拒绝。 $\|\mathbf{p}_i^{\mathrm{c}} - \mathbf{T} \oplus \mathbf{p}_j^{\mathrm{r}}\| > \epsilon_d$
- 两点的曲率之差距大于阈值，则拒绝。 $|\log \sigma_i^{\mathrm{c}} - \log \sigma_j^{\mathrm{r}}| > \epsilon_\sigma$
- 两点的法向量角度之差大于阈值，则拒绝。 $\mathbf{n}_i^{\mathrm{c}} \cdot \mathbf{T} \oplus \mathbf{n}_j^{\mathrm{r}} < \epsilon_n$

c) 算法流程：
1. 计算激光里程计当前帧和参考帧中每个点的曲率和法向量。
2. 将当前帧点云，经第 k 次估计的变换矩阵 T_k 变换到参考帧坐标系下，寻找最近点，然后根据上述点匹配规则来选择匹配点对
3. 根据筛选下来的匹配点对构建前述的目标损失函数，使用 LM 等优化算法，不断优化目标损失函数，直到收敛。

总结 NICP：
1. NICP 虽然同 ICP 一样也采用点到点匹配，但在误差项中额外加入了法向量距离，从而让优化更注重对角度的估计，使得位姿估计结果更加准确。
2. 利用更多的信息(点间距，曲率差，法向量差)来排除错误匹配，从而提高估计精度和迭代速度。

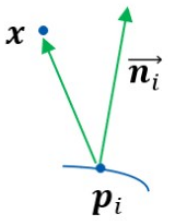
IMLS-ICP (Implicit Moving Least Square ICP) 基于隐函数的迭代最近点法
参考文献[4]，[5]

激光进行扫描时，由于采样差异，未必会扫到同一个点，所以我们从 ICP 点到点的匹配拓展到了 PL-ICP 点到线的匹配。但是激光在扫描时，也未必会扫描到之前帧扫到的线上，而是更有可能扫到之前帧扫到的面上。所以与其使用点到线的距离，使用点到面的距离来表示两帧点云的差异是更合理的一件事情。那么有没有点到面的匹配呢? 有的，IMLS-ICP 就是。
IMLS-ICP 主要提出了两点， 一个使用隐函数来表征点到面的距离，另一点是只选择对于各个姿态影响较大(可观)的激光点参与匹配点面匹配，而不是所有点

a) 使用隐函数来表征点到面的距离
在 IMLS-ICP 方法使用了 SDF(Signed Distance Function, 符号距离函数)来表示点 x 到曲面的距离。

$$I^{P_k}(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{p}_i \in P_k} \left(W_{\boldsymbol{p}_i}(x) \left((\boldsymbol{x} - \boldsymbol{p}_i) \cdot \vec{\boldsymbol{n}_i} \right) \right)}{\sum_{\boldsymbol{p}_j \in P_k} W_{\boldsymbol{p}_j}(x)}$$

这里:
p_i 为前一帧激光点云 P_k 中的点
n_l 为点 p_i 处的单位法向量
$W_{pi}(x) = \exp(-\|x - p_i\|^2/h^2)$ 代表点 p_i 的权重，显然离 x 越近的点权重越大。并且 $W_{pi}(x)$在 x 距离 p_i 较远时，会迅速减小，当距离为 3h 时，权重基本为 0 了，所以我们只考虑离 x 足够近的点 p_i 。我们可以选择只计算离 x 的距离为 h 的球体内的所有点。(这里又可以使用 kd 树来加快搜索最近点)
$I^{pk}(x)$ 就代表了点 x 到点云集 P_k 构成的曲面的距离。至于为什么 $I^{pk}(x)$代表距离，示意图如下所示，非常好理解。



有了点面距离的定义，现在我们的目的就是不断调整 R 和 t，使得下面这个由所有匹配点之间点面距离构成的损失函数最小:

$$min \sum_{\boldsymbol{x}_j \in S_k} \left| I^{P_k}(\boldsymbol{R}\boldsymbol{x}_j + \boldsymbol{t}) \right|$$

其中 x_j 代表新一帧激光点云 S_k 中的任意一点。

为了更好地优化上述目标函数，我们可以换一种表达。假如我们可以找到点 x_j 在上一帧曲面上的投影 y_j 的话，上述目标函数就可以简化为:

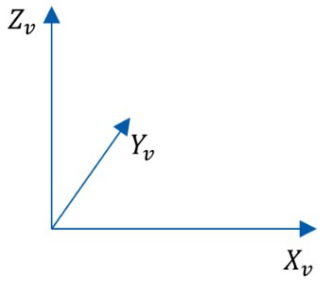
$$min \sum_{\boldsymbol{x}_j \in S_k} \left| \vec{\boldsymbol{n}_j} \cdot (\boldsymbol{R}\boldsymbol{x}_j + \boldsymbol{t} - \boldsymbol{y}_j) \right|^2$$

其中投影点 y_j 可以根据下面这个约束求得:

$$\boldsymbol{y}_j = \boldsymbol{x}_j - I^{P_k}(\boldsymbol{x}_j)\vec{\boldsymbol{n}_j}$$

其中 n_J 表示投影点 y_j 处的法向量。$I^{pk}(x_j)$代表点 x_j 到曲面的符号距离。
但是投影点 y_j 处的法向量如何求呢? 我们可以将前一帧激光点云中距离点 x_j 最近的一个点的法向量作为投影点 y_j 的法向量。这是一种合理的近似。
至此，由点到面的距离构成的目标损失函数就有了。

b) 代表点(Informative Point)选取
在 IMLS-ICP 方法中，并不选择所有点参与匹配，而是选择那些对于各个姿态影响较大(可观)的激光点参与匹配点面匹配，这样既可以提高匹配精准度，又可以提高计算效率。代表点的选取具体遵循以下原则:



1. 该点为结构化的点，即具有良好的曲率和法向量定义。曲率越小，代表该点越平直，也代表该点的法相量定义越好。
2. 保证选择激光点的均衡以保持可观性。
保持 3 个平移方向(X_v, Y_v, Z_v)的可观性:

$$a_{2D}^2|\overrightarrow{n_l}\cdot X_v|$$
$$a_{2D}^2|\overrightarrow{n_l}\cdot Y_v|$$
$$a_{2D}^2|\overrightarrow{n_l}\cdot Z_v|$$

比如可以使用
$a_{2D}^2|\overrightarrow{n_l}\cdot X_v|$筛选出法向量与 X_v 轴夹角尽可能小的点，相当于小车两侧平面上的激光点。
同理:
$a_{2D}^2|\overrightarrow{n_l}\cdot Y_v|$筛选出法向量与 Y_v 轴夹角尽可能小的点，相当于小车的正前方垂直于地面的平面上的激光点。

$a_{2D}^2|\overrightarrow{n_l}\cdot Z_v|$ 筛选出法向量与 Z_v 轴夹角尽可能小的点，相当于地面上的激光点。
同理，我们可以保持 3 个姿态角(roll, pitch, yaw)的可观性:

$$a_{2D}^2(x_i\times\overrightarrow{n_l})\cdot X_v$$
$$-a_{2D}^2(x_i\times\overrightarrow{n_l})\cdot X_v$$
$$a_{2D}^2(x_i\times\overrightarrow{n_l})\cdot Y_v$$
$$-a_{2D}^2(x_i\times\overrightarrow{n_l})\cdot Y_v$$
$$a_{2D}^2(x_i\times\overrightarrow{n_l})\cdot Z_v$$
$$-a_{2D}^2(x_i\times\overrightarrow{n_l})\cdot Z_v$$

综上，IMLS-ICP 由于实现的点到面的匹配，所以是比 ICP，PL-ICP 更为鲁棒和精确的点云匹配方法，而且融合了工程上如何筛选匹配点的策略，使得算法更为高效。

参考文献:

[1] Least-Squares Fitting of Two 3-D Points Sets, K.S. ARUN, T.S. HUANG, S.D. BLOSTEIN.

[2] An ICP variant using a point-to-line metric, Andrea Censi

[3] NICP: Dense Normal Based Point Cloud Registration, Jacopo Serafin and Giorgio Grisetti

[4] Provably Good Moving Least Squares, Ravikrishna Kolluri

[5] IMLS-SLAM: scan-to-model matching based on 3D data, Jean-Emmanuel Deschaud