ICP点云匹配报告

本报告力图读取10 个二维激光点云数据，通过迭代最近点（ICP）算法及其优化的方式融合出一整张点云地图，并由此得到机器人在每一帧上的位置和姿态信息，以得出其在地图中的运动轨迹。其最终结果好坏由MATLAB中的对等函数pcregistericp()处理。

报告利用两种匹配方法，同时阐释了一废案。

原初思路：相邻帧点到点匹配

算法文件中，icp_mine_no_acc.m的思路非常简单，利用了最原初的点到点方法，匹配相邻两帧的点云地图。其基本思路如下：

- **最近邻匹配**：使用最近邻搜索 `knnsearch(Q, P)` 为每个 `P` 中点找到 `Q` 中最近点。其中，`P` 为待匹配点云位置集，取当前帧；`Q` 为参考位置集，取上一帧。在实操中，图方便可以采用暴力搜索：

$$p_{match} = p_k, d_k = \underset{i \in P, j \in Q}{min}(\|p_i - q_j\|)$$

> 也可以利用K维树(KD-Tree)加快搜索。MATLAB中默认采用后者，但经过试验其实前者没慢多少。

- **刚体变换估计**：最经典的Kabsch算法。

$$\bar{\mathbf{p}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{p}_i, \quad \bar{\mathbf{q}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{q}_i$$

$$\mathbf{P}' = \mathbf{P} - \bar{\mathbf{p}}, \quad \mathbf{Q}' = \mathbf{Q} - \bar{\mathbf{q}}$$

构造协方差矩阵：

$$\mathbf{H} = \mathbf{P}'^{\top}\mathbf{Q}'$$

奇异值分解得旋转矩阵$\mathbf{R}$：

$$\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}$$

$$\mathbf{R} = \mathbf{V}\mathbf{U}^{\top}$$

要注意的是，有的时候$\mathbf{V}$会得出镜像结果（ $\det(\mathbf{R}) < 0$），因此需修正反射：

$$\mathbf{V}[:,3] = -\mathbf{V}[:,3], \quad \mathbf{R} = \mathbf{V}\mathbf{U}^\top$$

- **配准误差判断**：最小二乘中使用均方根误差（RMSE），若前后误差变化小于容限 `tolerance`，认为已收敛。
- **变换累积**：通过单帧变换矩阵 `T_icp = T_step * T_icp` 将每一帧变换累积（即当前帧相对于初始帧的变换）。其组成显然为：

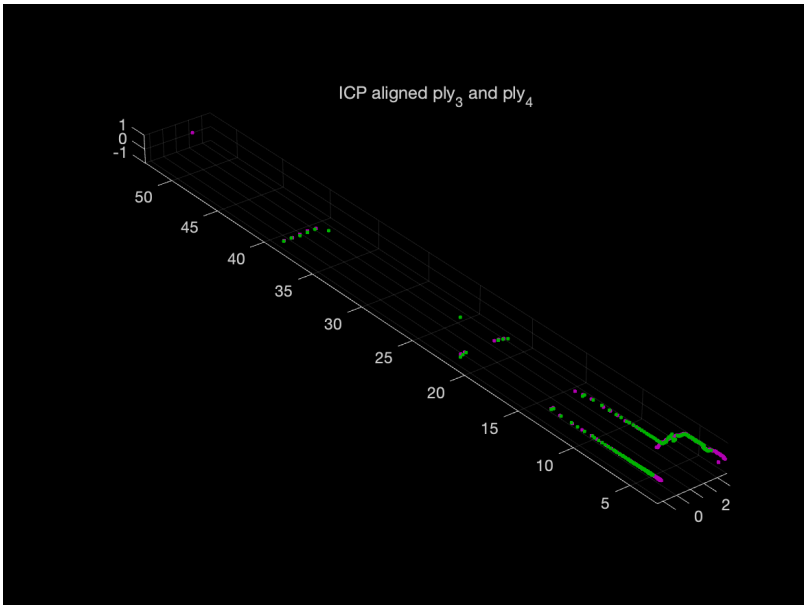
$$\mathbf{T}_{icp} = \begin{pmatrix} \mathbf{R} & \mathbf{p} \\ 0 & 1 \end{pmatrix}$$

- **点云融合**：`pcmerge` 加和，用于构建完整地图。
- **轨迹绘制**：每一帧整体相对最初帧的变换矩阵$\mathbf{T_k}$可迭代得到：

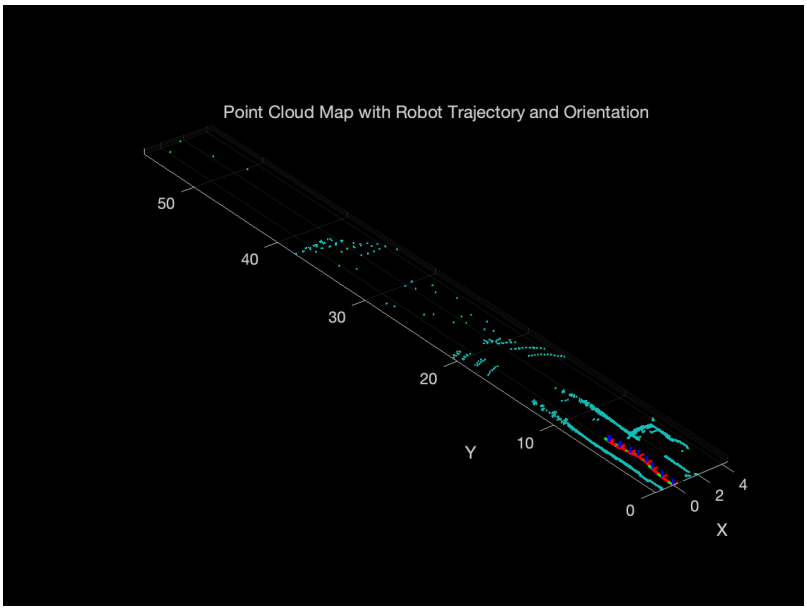
$$\mathbf{T_k} = \prod_{0}^{k} \mathbf{T}_{icp_i}$$

> 因此，可以将每帧的变换中的平移向量提取出来并绘制轨迹。

经此算法，对于两帧结构都有板有眼的相邻点云，光看每一帧的匹配结果，似乎还较令人满意。（结果存于ICP_results中）



然而，当把他们全都加起来形成整体点云的时候……

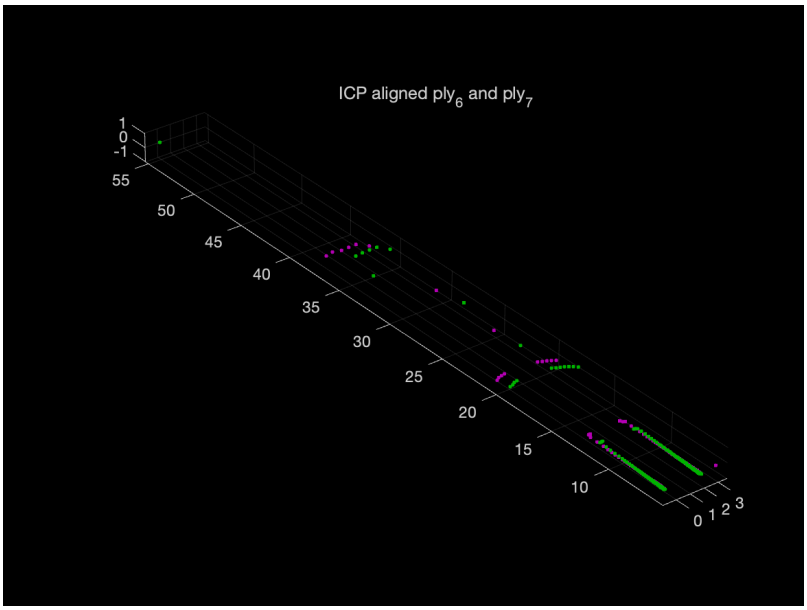


> *点云中间的线和坐标为机器人位姿*

会发现远处的八字和一字居然没有对齐。这并非孤例，经过试验，MATLAB自带的pcregistericp()在相邻帧匹配时的表现**还要糟糕**。这可由两者的轨迹匹配误差得知：

```
>> RMSE
RMSE between estimated and true positions: 0.3641 meters
Maximum position error: 0.7067 meters
```

虽然达成了题目要求（小于两米），但是极为不着调。经复盘，其问题出现在机器人离开右墙角的后几帧：



原因不难理解。对人类而言，我们看到的是两段迥异的特征段，然而点对点的匹配方法使机器人只觉得要关注点云最密集的那一段，也即近端。因此，点稀疏却至关重要的八字部分和一字部分

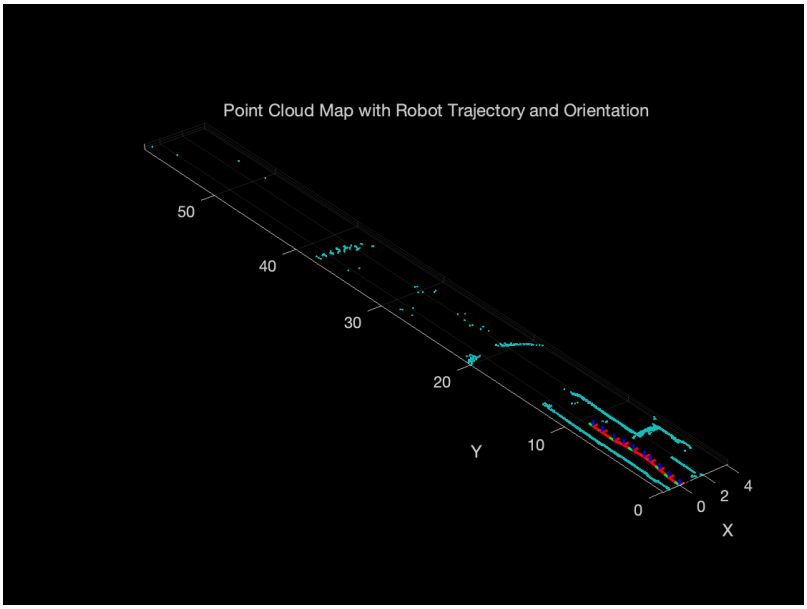
就被自动让位给了近端，是而产生错配。因此，自然能够想到：要是我把之前的点云全部叠加起来，让八字部分变密集，是否就能解决该问题?

解决八字不匹配：累加点对点匹配

答案是：是的，确实可以。实现该算法，只需要在上一算法的基础上，先对待匹配点云进行总变换，使得他得到和相邻帧匹配时相对上一帧一样的位姿，而不是仍然挤在原点（事实证明，后者会导致严重的错配）：

$$\mathbf{P'_k} = \mathbf{T_{k-1}} \mathbf{P_k}$$

参考点云采用总和点云即可。结果如下（存储在ICP_result_acc中）：



可以看到，整体的对齐有了极大的改善。pcregistericp()自己也支持这样的对齐方法，与其相比，差距为：

```
>> RMSE
RMSE between estimated and true positions: 0.0402 meters
Maximum position error: 0.0817 meters
```

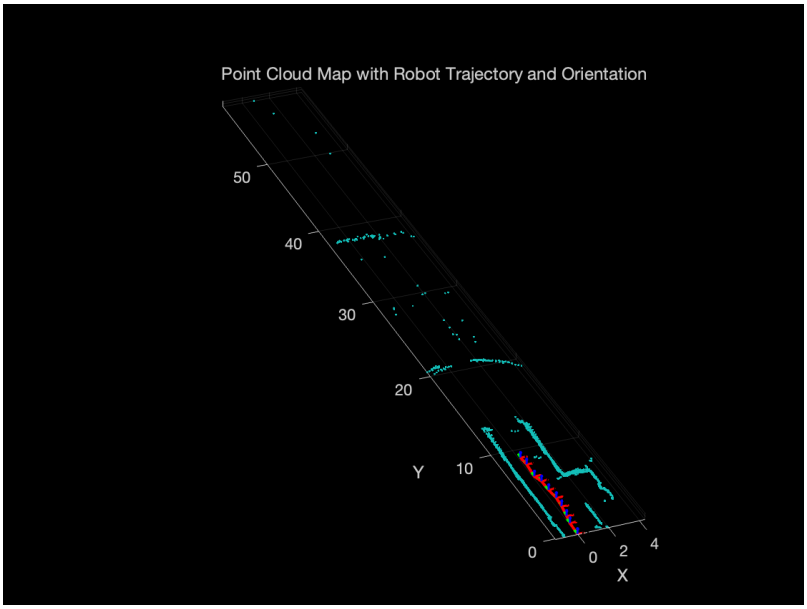
被大大减小，因而成功。当然，需要注意的是，累加而成的$\mathbf{T}_k$本身具有一定舍入误差，纯靠点云定位最后说不定也会越来越偏移真实图像，此未可知。

一个废案：点对线匹配

这里简单说一个为解决相邻帧匹配误差问题而成的废案。既然这个图像如此横平竖直，与其最小化点与点之间的距离，最小化点与点的法线方向距离是否就能让八字之间匹配起来？于是，目标函数改为：

$$\min_{R,t} \sum_i \left(\mathbf{n}_i^\top (R\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i) \right)^2$$

其中 $\mathbf{n}_i$ 为目标点云中 $\mathbf{q}_i$ 对应的法向量。
事实上，确实可以，它解决了大部分帧八字不对齐的问题。然而这样的匹配方式使得某些帧陷入了局部最优解，匹配出了一些极为猎奇的图样（没存，请君想象）。而如果采用剔除一些误匹配点的方式，确实可以修正这一问题，但是也导致精度不尽人意。因此，本方案最终被放弃了，只留下了最终也把他累加起来匹配的尝试（结果在ICP_result_p2l中）：




```
>> RMSE
RMSE between estimated and true positions: 0.1990 meters
Maximum position error: 0.3173 meters
```

> *差距不大，但不能说没有*

尾注

- icp_mine_acc.m为第二种算法，也是评判的采信算法。
- 如不信本报告中的数据，可以自行查看附赠的robot_tf_acc.mat等机器人转换矩阵数据，并和robot_tf_true_a.mat等用MATLAB包函数得到的值做对比。还附赠了一个RMSE.m用来计算rmse。
- 代码中只有必要注释，显示了调试过程。祝看懂。