这是标题

陈烁龙

2022 年 9 月 21 日

目录

插图

表格

1 概述

本次论文为：《Targetless Calibration of LiDAR-IMU System Based on Continuous-time Batch Estimation》，其是一篇关于传感器标定的文章。其使用 B 样条曲线，在连续时间上对传感器的轨迹进行建模 (主要是 IMU 和激光雷达)。其不需要靶标，适应的场景较多。

2 前置知识

2.1 贝塞尔曲线

贝塞尔曲线主要用于二维图形应用程序中的数学曲线，曲线由起始点，终止点 (也称锚点) 和控制点组成，通过调整控制点，通过一定方式绘制的贝塞尔曲线形状会发生变化[1]。

贝塞尔曲线的性质：曲线总是通过第一个点和最后一个点；第一个点的切线即为第一个点和第二个点的连线；最后一个点的切线即为最后一个点和倒数第二个点的连线。

对于 n 次的贝塞尔曲线，其有 $n+1$ 个控制点，曲线上点的位置由控制点和参数 $t \in [0,1]$ 决定。其公式为：

$$B_n(t) = \sum_{i=0}^{n} C_n^i (1-t)^{n-i} t^i P_i \quad t \in [0,1]$$

其中：

$$C_n^i = \frac{n!}{(n-i)! \cdot i!}$$

例如，对于 3 次的贝塞尔曲线，其公式为：

$$\begin{aligned} B_3(t) =&(1-t)^3 P_0 + 3t(1-t)^2 P_1 \\ &+ 3t^2(1-t)P_2 + t^3 P_3 \end{aligned}$$

另外，对于计算机适合的计算贝塞尔曲线递推方法为：

$$\begin{aligned} &B_n(t|P_0, P_1, \ldots, P_n) = \\ &(1-t)B_{n-1}(t|P_0, P_1, \ldots, P_{n-1}) \\ &+ tB_{n-1}(t|P, P_1, \ldots, P_n) \end{aligned}$$

比如，同样对于 3 阶的贝塞尔曲线，其公式为：

$$\begin{cases} B_3(t|P_0,P_1,P_2,P_3) =(1-t)B_2(t|P_0,P_1,P_2) \\ \qquad\qquad + tB_2(t|P_1,P_2,P_3) \\ B_2(t|P_0,P_1,P_2) =(1-t)B_1(t|P_0,P_1) \\ \qquad\qquad + tB_1(t|P_1,P_2) \\ B_2(t|P_1,P_2,P_3) =(1-t)B_1(t|P_1,P_2) \\ \qquad\qquad + tB_1(t|P_2,P_3) \\ B_1(t|P_i,P_j) =(1-t)B_0(t|P_i) + tB_0(t|P_j) \\ \qquad\qquad =(1-t)P_i + tP_j \end{cases}$$

贝塞尔曲线如下图所示：

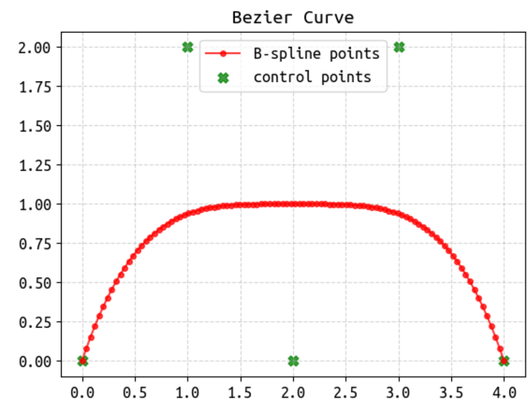


图 1: 贝塞尔曲线

2.2 B 样条曲线

B 样条是通过一组指定点集而生成平滑曲线的柔性带。简单地说，B 样条曲线就是通过控制点局部控制形状的曲线[2]。

对于 k 次[3]的 B 样条曲线，其有 $n+1$ 个控制点、$m+1=(n+k+1)+1$ 个节点，即：$t_0 < \cdots < t_m$，且每个 $t \in [t_i, t_{i+1}]$ 节点处的值只会由 $k+1$ 个控制点 $P_i, P_{i+1}, \cdots, P_{i+k}$ 约束，其公式为：

$$B_{n,k}(t) = \sum_{i=0}^{n} P_i \cdot b_{i,k}(t)$$

[1]注意是整体曲线，这和后面的 B 样条曲线不同。

[2]注意是局部控制，因为其多项式的次数 (k) 独立于控制点的数目 $(n+1)$，但是有一定的限制。

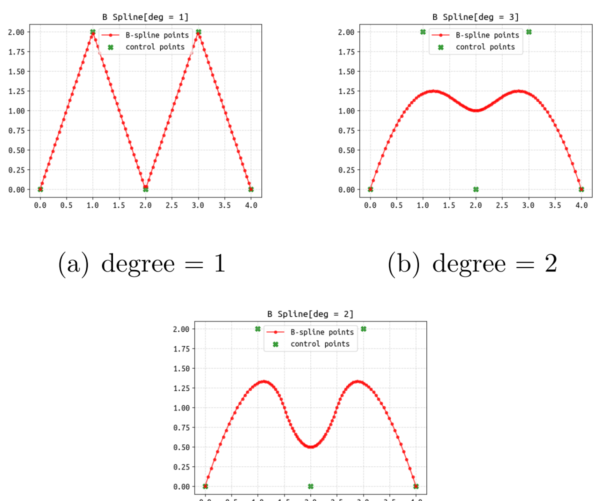
[3]阶数等于次数加 1。

其中 $b_{i,n}(t)$ 可以通过 de Boor 递归公式得到：

$$b_{i,0}(t)=\begin{cases}1 & t_i\le t\le t_{i+1}\\ 0 & \dots\end{cases}$$

$$b_{i,k}(t)=\frac{t-t_i}{t_{i+k}-t_i}b_{i,k-1}(t)+\frac{t_{i+k+1}-t}{t_{i+k+1}-t_{i+1}}b_{i+1,k-1}(t)$$

当节点等距，称 B 样条为均匀。如果节点的重复度为 $k+1$(即 $t_0=\cdots=t_k, t_{m-k}=\cdots=t_m$) 时，为准 B 样条均匀。



(a) degree = 1　(b) degree = 2

(c) degree = 3

图 2: 准 B 样条曲线

3 论文算法

3.1 位姿 B 样条

对于一个均匀 B 样条曲线，我们可以通过矩阵的形式来表达它：

$$\boldsymbol{p}(t)=\sum_{j=0}^{d}\boldsymbol{u}^T\boldsymbol{M}_{(j)}^{(d+1)}\boldsymbol{p}_{i+j}$$

其中 d 为该均匀 B 样条曲线的次，也就是说，对于 $t\in[t_i,t_{i+1}]$ 时间范围内点点，其只会受控制点 $\{\boldsymbol{p}_i,\dots,\boldsymbol{p}_{i+d}\}$ 的影响，论文中选取 $d=3$。公式中的向量和矩阵为：

$$\boldsymbol{M}^{(3+1)}=\frac{1}{6}\begin{pmatrix}1&4&1&0\\-3&0&3&0\\3&-6&3&0\\-1&3&-3&1\end{pmatrix}\quad \boldsymbol{u}=\begin{pmatrix}1\\u\\u^2\\u^3\end{pmatrix}$$

$$u=\frac{t-t_i}{t_{i+1}-t_i}$$

符号 $\boldsymbol{M}_{(j)}^{(3+1)}$ 表示矩阵 $\boldsymbol{M}^{(3+1)}$ 的第 j 列向量。通过推导，可以得到该样条曲线对时间的函数：

$$\begin{cases}\dot{\boldsymbol{p}}(t)=\sum_{j=0}^{d}\frac{1}{t_{i+1}-t_i}\begin{pmatrix}0&1&2u&3u^2\end{pmatrix}\boldsymbol{M}_{(j)}^{(d+1)}\boldsymbol{p}_{i+j}\\ \ddot{\boldsymbol{p}}(t)=\sum_{j=0}^{d}\frac{1}{(t_{i+1}-t_i)^2}\begin{pmatrix}0&0&2&6u\end{pmatrix}\boldsymbol{M}_{(j)}^{(d+1)}\boldsymbol{p}_{i+j}\end{cases}$$

上面的 B 样条曲线可以描述位姿量中的平移量。

对于位姿量中的旋转量，还需要进行相应的变换。不难得到，下面形式的 B 样条等价于上文中的公式：

$$\boldsymbol{p}(t)=\boldsymbol{p}_i+\sum_{j=1}^{d}\boldsymbol{u}^T\tilde{\boldsymbol{M}}_{(j)}^{(d+1)}(\boldsymbol{p}_{i+j}-\boldsymbol{p}_{i+j+1})$$

d 当 $d=3$ 时，公式中矩阵 $\tilde{\boldsymbol{M}}^{(3+1)}$ 为：

$$\tilde{\boldsymbol{M}}^{(3+1)}=\frac{1}{6}\begin{pmatrix}6&5&1&0\\0&3&3&0\\0&-3&3&0\\0&1&-2&1\end{pmatrix}$$

我们利用四元素表示旋转，则有：

$$\boldsymbol{q}(t)=\boldsymbol{q}_i\otimes\prod_{j=1}^{d}\exp(\boldsymbol{u}^T\tilde{\boldsymbol{M}}_{(j)}^{(d+1)}\log(\boldsymbol{q}_{i+j+1}^{-1}\otimes\boldsymbol{q}_{i+j})$$

其中，$\exp(\cdot)$ 表示从李代数到姿态流形的指数映射，而 $\log(\cdot)$ 表示从姿态流形到李代数的对数映射，二者是相反的过程。

通过对旋转 B 样条曲线和位移 B 样条曲线求时间微分，可以轻松的将其和 IMU 的输出角速度和线速度对应上：

$$\begin{cases}{}^{I}\boldsymbol{a}(t)={}^{I_0}_{I}\boldsymbol{R}^T(t)({}^{I_0}_{I}\ddot{\boldsymbol{p}}(t)-{}^{I_0}\boldsymbol{g})\\ {}^{I}\boldsymbol{\omega}(t)={}^{I_0}_{I}\boldsymbol{R}^T(t){}^{I_0}_{I}\dot{\boldsymbol{R}}(t)\end{cases}$$

其中 ${}^{I_0}\boldsymbol{g}$ 表示 IMU 参考帧 (即第一帧) 上的重力。

3.2 流程

首先，我们可以通过 IMU 的输出，拟合一条姿态 B 样条曲线[4]，其最小二乘问题定义为：

$$\boldsymbol{q}_0, \cdots, \boldsymbol{q}_N = \min \sum_{k=0}^{M} = \| {}^{I_k}\boldsymbol{\omega}_m - {}^{I_0}_{I}\boldsymbol{R}^T(t_k) {}^{I_0}_{I}\dot{\boldsymbol{R}}^T(t_k) \|$$

其中 ${}^{I_k}\boldsymbol{\omega}_m$ 为 t_k 时刻陀螺的角速度输出。注意，在优化的时候，${}^{I_0}_{I}\boldsymbol{q}(t_0)$ 保持为单位四元素[5]。之后，我们就可以计算任意时刻的 IMU 位姿了。

同时，我们基于 NDT 点云注册方法，进行激光里程计的估计，得到帧间的位姿变化 ${}^{L_k}_{L_{k+1}}\boldsymbol{q}$。当然，我们可以得到在这一段时间内 IMU 的位姿变化 ${}^{I_k}_{I_{k+1}}\boldsymbol{q}$。显然，这两个位姿变换之间满足：

$${}^{I_k}_{I_{k+1}}\boldsymbol{q} \otimes {}^{I}_{L}\boldsymbol{q} = {}^{I}_{L}\boldsymbol{q} \otimes {}^{L_k}_{L_{k+1}}\boldsymbol{q}$$

进一步，有：

$$\alpha_k \left(\left[{}^{I_k}_{I_{k+1}}\boldsymbol{q} \right]_L - \left[{}^{L_k}_{L_{k+1}}\boldsymbol{q} \right]_L \right) {}^{I}_{L}\boldsymbol{q} = \boldsymbol{0}$$

其中：$[\cdot]_L$ 表示四元素左乘积矩阵[6]，$[\cdot]_R$ 表示四元素右乘积矩阵。而 α_k 是一个权因子：

$$\begin{cases} r_k = \|2(\arccos({}^{I_k}_{I_{k+1}}q_w) - \arccos({}^{L_k}_{L_{k+1}}q_w)) \\ \alpha_k = \begin{cases} 1 & r_k < threshold \\ \dfrac{threshold}{r_k} & otherwise \end{cases} \end{cases}$$

[4]参数其实就是得到曲线上的控制点，因为控制点确定，曲线方程就确定了。

[5]类似与旋转矩阵中的单位矩阵，代表不旋转。

[6] 假设有两个四元素 $\boldsymbol{q} = \begin{pmatrix} q_w & \boldsymbol{q}_s^T \end{pmatrix}^T$、$\boldsymbol{p} = \begin{pmatrix} p_w & \boldsymbol{p}_s^T \end{pmatrix}^T$，那么 $\boldsymbol{p} \otimes \boldsymbol{q} = [\boldsymbol{p}]_L \otimes \boldsymbol{q} = [\boldsymbol{q}]_R \otimes \boldsymbol{p}$。其中：

$$[\boldsymbol{p}]_L = \begin{pmatrix} p_w & -\boldsymbol{p}_s^T \\ \boldsymbol{p}_s & p_w\boldsymbol{I} + \lfloor \boldsymbol{p}_s \rfloor_\times \end{pmatrix} \quad [\boldsymbol{q}]_R = \begin{pmatrix} q_w & -\boldsymbol{q}_s^T \\ \boldsymbol{q}_s & q_w\boldsymbol{I} - \lfloor \boldsymbol{q}_s \rfloor_\times \end{pmatrix}$$

所以，对于多个对应段内的姿态变换，我们可以列出方程组，求解得到激光雷达到 IMU 的姿态变换 ${}^{I}_{L}\boldsymbol{q}$。至于初始化时候的 IMU 和激光雷达的位移量 ${}^{I}\boldsymbol{p}_L$，由于加速度和重力耦合，所以不利于估计。在论文中不初始化位移量。

初始化了 IMU 和 LiDAR 以及他们之间的姿态变换关系之后，就可以利用 IMU 的连续时间姿态 B 样条曲线对 LiDAR 帧进行去畸变操作，这样可以改善 LiDAR 的里程计精度。随后，论文将初步得到的 LiDAR 点云地图进行分块，然后计算每一个快内的点均值向量和方差矩阵。对于反差矩阵，可以求解得到其三个特征值：$\lambda_0 \leq \lambda_1 \leq \lambda_2$，并构造一个量来描述该块内点云的平面度 (分布趋于平面的程度)：

$$\mathcal{P} = 2\frac{\lambda_1 - \lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}$$

$\mathcal{P}$ 越接近 1，越像平面。对于平面度大于阈值的平面，我们对其点云进行基于 RANSAC 的平面拟合，得到平面参数：$\pi = \left[\boldsymbol{n}^T, d\right]^T$。其中 $\boldsymbol{n}$ 是平面的单位法向量，d 是坐标原点到平面的距离。

完成了初始化和 LiDAR 点云面片化，我们就可以进行图优化了。我们定义状态向量为：

$$\boldsymbol{x} = \left[{}^{I}_{L}\boldsymbol{q}^T, {}^{I}\boldsymbol{p}_L^T, \boldsymbol{x}_q^T, \boldsymbol{x}_p^T, \boldsymbol{b}_a^T, \boldsymbol{b}_g^T \right]^T$$

其中，前两个参数向量为 IMU 和 LiDAR 之间的位姿关系，后两个参数为 B 样条曲线的控制点位姿，最后两个为 IMU 的零偏误差。

所以，对于一系列的 LiDAR 点云帧 $\mathcal{L}$、加速度计的输出 $\mathcal{A}$、角速度计的输出 $\mathcal{W}$，可以定义一个最小二乘问题：

$$\hat{\boldsymbol{x}} = \min \left\{ \sum_{k=\mathcal{A}} \|\boldsymbol{r}_a^k\|^2_{\boldsymbol{\Sigma}_a} + \sum_{k=\mathcal{W}} \|\boldsymbol{r}_\omega^k\|^2_{\boldsymbol{\Sigma}_\omega} + \sum_{j=\mathcal{L}} \|\boldsymbol{r}_l^j\|^2_{\boldsymbol{\Sigma}_l} \right\}$$

其中：

$$\begin{cases} \boldsymbol{r}_a^k = {}^{I_k}\boldsymbol{a}_m - {}^{I}\boldsymbol{a}(t_k) - \boldsymbol{b}_a \\ \boldsymbol{r}_\omega^k = {}^{I_k}\boldsymbol{\omega}_m - {}^{I}\boldsymbol{\omega}(t_k) - \boldsymbol{b}_g \end{cases}$$

$$\begin{cases} {}^{L_0}\boldsymbol{p}_{L_j} = {}_L^I\boldsymbol{R}^{T\,I_0}_{\,I_j}\boldsymbol{R}^I\boldsymbol{p}_L + {}_L^I\boldsymbol{R}^{T\,I_0}\boldsymbol{p}_{I_j} - {}_L^I\boldsymbol{R}^{T\,I}\boldsymbol{p}_L \\ {}^{L_0}\boldsymbol{p}_i = {}_L^I\boldsymbol{R}^{T\,I_0}_{\,I_j}\boldsymbol{R}_L^I\boldsymbol{R}^{L_j}\boldsymbol{p}_i + {}^{L_0}\boldsymbol{p}_{L_j} \\ \boldsymbol{r}_l^j = \left[{}^{L_0}\boldsymbol{p}_i^{\ T}, 1\right]\pi_j \end{cases}$$

推导：我们将第 j 帧点云里的第 i 个点 ${}^{L_j}\boldsymbol{p}_i$，首先推到同时刻的 IMU 坐标系里，然后再转到初始时刻坐标系里，最后再转回初始时刻 LiDAR 坐标系里，这样就相当与引入了 IMU 和 LiDAR 之间的外参。具体来说：

$${}^{L_0}\boldsymbol{p}_i = {}_L^I\boldsymbol{T}^{-1\,I_0}_{\ \ \,I_j}\boldsymbol{T}_L^I\boldsymbol{T}^{L_j}\boldsymbol{p}_i$$

其中：

$${}_L^I\boldsymbol{T} = \begin{pmatrix} {}_L^I\boldsymbol{R} & {}^L\boldsymbol{p}_L \\ \boldsymbol{0} & 1 \end{pmatrix} \quad {}_L^I\boldsymbol{T}^{-1} = \begin{pmatrix} {}_L^I\boldsymbol{R}^{-1} & -{}_L^I\boldsymbol{R}^{-1\,L}\boldsymbol{p}_L \\ \boldsymbol{0} & 1 \end{pmatrix}$$

通过推导可以得到上面的公式 (姿态和位移分开表达)。

经过批量优化之后，待估的状态就会变得更加精确。所以，之后就可以利用该状态再次进行 LiDAR 的点云畸变处理，而后重新优化小面片。可以通过多次迭代改善待估的状态精度。