武漢大學

River

姓名：陈烁龙

学号：2023202140019

学院：测绘学院

2024 年 5 月 8 日

目录

1 IMU 因子 **1**

2 Radar 因子 **1**

2.1 Radar Static Measurement (V1) 1

2.2 Radar Static Measurement (V2) 2

2.3 因子构建 2

3 初始化 **2**

4 因子加权 **3**

5 基于卡尔曼滤波器的样条先验 **4**

5.1 零偏先验 4

5.2 速度先验 4

6 边缘化 **4**

6.1 ceres 四元数求导相关问题 4

6.2 ceres 评估函数 5

6.3 舒尔补 7

6.4 测试案例 8

7 速度可视化 **9**

8 样条解析求导 **9**

9 ACKNOWLEDGMENT **11**

插图

表格

1 IMU 因子

对于 **IMU**，其理想输出和位姿轨迹之间的关系为：

$$\begin{cases} {}^{b}\boldsymbol{a}(t) = {}^{b_0}_{b}\boldsymbol{R}^{\top}(t)\cdot\left({}^{b_0}\ddot{\boldsymbol{p}}_I(t) - {}^{b_0}\boldsymbol{g}\right) \\ {}^{b}\boldsymbol{\omega}(t) = {}^{b_0}_{b}\boldsymbol{R}^{\top}(t)\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(t) \end{cases} \tag{1}$$

现考虑 **IMU** 零偏，则其输出为：

$$\begin{cases} {}^{b}\hat{\boldsymbol{a}}(\tau) = {}^{b}\boldsymbol{a}(\tau) + \boldsymbol{b}_a(\tau) + \boldsymbol{n}_a(\tau) \\ {}^{b}\hat{\boldsymbol{\omega}}(\tau) = {}^{b}\boldsymbol{\omega}(\tau) + \boldsymbol{b}_g(\tau) + \boldsymbol{n}_g(\tau) \end{cases} \tag{2}$$

其中 τ 时刻的零偏 $\boldsymbol{b}_\omega(\tau)$ 和 $\boldsymbol{b}_a(\tau)$ 可以通过相应的零偏样条计算得到。通过构建对应的因子，可以优化“旋转样条”、“速度样条”、“加速度计零偏样条”、“陀螺仪零偏样条”、“首帧下的重力向量”。

2 Radar 因子

2.1 Radar Static Measurement (V1)

The continuous-time trajectory is the one of other sensor (e.g., IMU), we have:

$${}^{b_0}\boldsymbol{p}_t = {}^{b_0}_{b}\boldsymbol{R}(\tau)\cdot{}^{b}\boldsymbol{p}_t(\tau) + {}^{b_0}\boldsymbol{p}_b(\tau) \tag{3}$$

with

$${}^{b}\boldsymbol{p}_t(\tau) = {}^{b}_{r}\boldsymbol{R}\cdot{}^{r}\boldsymbol{p}_t(\tau) + {}^{b}\boldsymbol{p}_r \tag{4}$$

then

$${}^{b_0}\boldsymbol{p}_t = {}^{b_0}_{b}\boldsymbol{R}(\tau)\cdot{}^{b}_{r}\boldsymbol{R}\cdot{}^{r}\boldsymbol{p}_t(\tau) + {}^{b_0}_{b}\boldsymbol{R}(\tau)\cdot{}^{b}\boldsymbol{p}_r + {}^{b_0}\boldsymbol{p}_b(\tau) \tag{5}$$

differentiate

$${}^{b_0}\dot{\boldsymbol{p}}_t = \boldsymbol{0}_{3\times1} = -\left[{}^{b_0}_{b}\boldsymbol{R}(\tau)\cdot{}^{b}_{r}\boldsymbol{R}\cdot{}^{r}\boldsymbol{p}_t(\tau)\right]_\times\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{b_0}_{b}\boldsymbol{R}(\tau)\cdot{}^{b}_{r}\boldsymbol{R}\cdot{}^{r}\dot{\boldsymbol{p}}_t(\tau) - \left[{}^{b_0}_{b}\boldsymbol{R}(\tau)\cdot{}^{b}\boldsymbol{p}_r\right]_\times\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{b_0}\dot{\boldsymbol{p}}_b(\tau) \tag{6}$$

$$\boldsymbol{0}_{3\times1} = -\left[{}^{b}_{r}\boldsymbol{R}\cdot{}^{r}\boldsymbol{p}_t(\tau)\right]_\times\cdot{}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{b}_{r}\boldsymbol{R}\cdot{}^{r}\dot{\boldsymbol{p}}_t(\tau) - \left[{}^{b}\boldsymbol{p}_r\right]_\times\cdot{}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}\dot{\boldsymbol{p}}_b(\tau) \tag{7}$$

$$\boldsymbol{0}_{3\times1} = -\left[{}^{r}\boldsymbol{p}_t(\tau)\right]_\times\cdot{}^{b}_{r}\boldsymbol{R}^{\top}\cdot{}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{r}\dot{\boldsymbol{p}}_t(\tau) - {}^{b}_{r}\boldsymbol{R}^{\top}\cdot\left[{}^{b}\boldsymbol{p}_r\right]_\times\cdot{}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{b}_{r}\boldsymbol{R}^{\top}\cdot{}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}\dot{\boldsymbol{p}}_b(\tau) \tag{8}$$

thus, the velocity of target $\{t\}$ with respect to the radar $\{r\}$ parameterized in $\{r\}$ could be expressed as:

$${}^{r}\dot{\boldsymbol{p}}_t(\tau) = \left[{}^{r}\boldsymbol{p}_t(\tau)\right]_\times\cdot{}^{b}_{r}\boldsymbol{R}^{\top}\cdot{}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{b}_{r}\boldsymbol{R}^{\top}\cdot\left[{}^{b}\boldsymbol{p}_r\right]_\times\cdot{}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) - {}^{b}_{r}\boldsymbol{R}^{\top}\cdot{}^{b_0}_{b}\boldsymbol{R}^{\top}(\tau)\cdot{}^{b_0}\dot{\boldsymbol{p}}_b(\tau) \tag{9}$$

2.2 Radar Static Measurement (V2)

Considering the extrinsics between the radar and the IMU, we have:

$$^{b_0}\boldsymbol{p}_r(\tau) = {}^{b_0}_{b}\boldsymbol{R}(\tau) \cdot {}^{b}\boldsymbol{p}_r + {}^{b_0}\boldsymbol{p}_b(\tau) \tag{10}$$

$$^{b_0}\dot{\boldsymbol{p}}_r(\tau) = -\left[{}^{b_0}_{b}\boldsymbol{R}(\tau) \cdot {}^{b}\boldsymbol{p}_r\right]_\times \cdot {}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{b_0}\dot{\boldsymbol{p}}_b(\tau) \tag{11}$$

thus, the velocity of radar $\{r\}$ with respect to the frame $\{b_0\}$ parameterized in $\{r\}$ could be expressed as:

$${}^{b}_{r}\boldsymbol{R}^\top \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot {}^{b_0}\dot{\boldsymbol{p}}_r(\tau) = {}^{b}_{r}\boldsymbol{R}^\top \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot \left(-\left[{}^{b_0}_{b}\boldsymbol{R}(\tau) \cdot {}^{b}\boldsymbol{p}_r\right]_\times \cdot {}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) + {}^{b_0}\dot{\boldsymbol{p}}_b(\tau)\right) \tag{12}$$

actually, by introducing ${}^{r}\boldsymbol{p}_t^\top(\tau)$ in (9) and considering (12), we have:

$$\begin{aligned} {}^{r}\boldsymbol{p}_t^\top(\tau) \cdot {}^{r}\dot{\boldsymbol{p}}_t(\tau) &= {}^{r}\boldsymbol{p}_t^\top(\tau) \cdot \left({}^{b}_{r}\boldsymbol{R}^\top \cdot \left[{}^{b}\boldsymbol{p}_r\right]_\times \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot {}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) - {}^{b}_{r}\boldsymbol{R}^\top \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot {}^{b_0}\dot{\boldsymbol{p}}_b(\tau)\right) \\ &= {}^{r}\boldsymbol{p}_t^\top(\tau) \cdot \left(-{}^{b}_{r}\boldsymbol{R}^\top \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot {}^{b_0}\dot{\boldsymbol{p}}_r(\tau)\right) \\ &= -{}^{r}\boldsymbol{p}_t^\top(\tau) \cdot {}^{b}_{r}\boldsymbol{R}^\top \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot {}^{b_0}\dot{\boldsymbol{p}}_r(\tau) \end{aligned} \tag{13}$$

thus, we have:

$${}^{r}\boldsymbol{p}_t^\top(\tau) \cdot {}^{b}_{r}\boldsymbol{R}^\top \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot {}^{b_0}\dot{\boldsymbol{p}}_r(\tau) = -{}^{r}\boldsymbol{p}_t^\top(\tau) \cdot {}^{r}\dot{\boldsymbol{p}}_t(\tau) \tag{14}$$

2.3 因子构建

对于 **radar** 量测有：

$$v = \frac{{}^{r}\boldsymbol{p}_t^\top \cdot {}^{r}\dot{\boldsymbol{p}}_t}{d} \quad \text{s.t. } {}^{r}\boldsymbol{p}_t = d \cdot \begin{pmatrix} \cos\theta\cos\phi \\ \sin\theta\cos\phi \\ \sin\phi \end{pmatrix} \tag{15}$$

当 **target** 是静态的时候，有：

$$v \cdot d = -{}^{r}\boldsymbol{p}_t^\top(\tau) \cdot {}^{b}_{r}\boldsymbol{R}^\top \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot {}^{b_0}\dot{\boldsymbol{p}}_r(\tau) \tag{16}$$

通过上式即可构建 **radar** 因子。

3 初始化

使用速度层面的预积分进行初始化。首先进行旋转样条的拟合以及每一帧 **radar** 数据 (包含多个 **target** 量测) 的速度计算。**radar** 帧速度计算通过最小二乘，计算得到 ${}^{b_0}\dot{\boldsymbol{p}}_r(\tau)$:

$$v \cdot d = -{}^{r}\boldsymbol{p}_t^\top(\tau) \cdot {}^{b}_{r}\boldsymbol{R}^\top \cdot {}^{b_0}_{b}\boldsymbol{R}^\top(\tau) \cdot {}^{b_0}\dot{\boldsymbol{p}}_r(\tau) \tag{17}$$

对于 **IMU** 量测，基于连续时间的预积分 (速度层面)，有：

$${}^{b}\boldsymbol{a}(t) = {}^{b_0}_{b}\boldsymbol{R}^\top(t) \cdot \left({}^{b_0}\ddot{\boldsymbol{p}}_b(t) - {}^{b_0}\boldsymbol{g}\right) \quad \rightarrow \quad {}^{b_0}_{b}\boldsymbol{R}(t) \cdot {}^{b}\boldsymbol{a}(t) = {}^{b_0}\ddot{\boldsymbol{p}}_b(t) - {}^{b_0}\boldsymbol{g} \tag{18}$$

在 $\tau \in [t_k, t_{k+1}]$ 时间段内，有：

$$\int_{t_k}^{t_{k+1}} {}^{b_0}_{b}\boldsymbol{R}(\tau) \cdot {}^{b}\boldsymbol{a}(\tau) \cdot d\tau = {}^{b_0}\dot{\boldsymbol{p}}_b(t_{k+1}) - {}^{b_0}\dot{\boldsymbol{p}}_b(t_k) - {}^{b_0}\boldsymbol{g} \cdot (t_{k+1} - t_k) \tag{19}$$

其中的 ${}^{b_0}\dot{\boldsymbol{p}}_b(t_{k+1})$ 和 ${}^{b_0}\dot{\boldsymbol{p}}_b(t_k)$ 可以通过 **(11)** 得到，也即：

$${}^{b_0}\dot{\boldsymbol{p}}_b(\tau) = {}^{b_0}\dot{\boldsymbol{p}}_r(\tau) + \left[{}^{b_0}_{b}\boldsymbol{R}(\tau) \cdot {}^{b}\boldsymbol{p}_r\right]_\times \cdot {}^{b_0}_{b}\dot{\boldsymbol{R}}(\tau) \tag{20}$$

至此，重力向量即可初始化完成。而后使用 **IMU** 因子、**radar** 因子对系统内的所有变量进行 **refinement**。完成后将 $\{b_0\}$ 的负 **z** 轴对齐到重力向量方向，将其作为世界参考系 $\{w\}$。

4 因子加权

IMU 量测建模为：

$$\begin{cases} {}^{b}\hat{\boldsymbol{a}}(\tau) = {}^{b}\boldsymbol{a}(\tau) + \boldsymbol{b}_a(\tau) + \boldsymbol{n}_a(\tau) \\ {}^{b}\hat{\boldsymbol{\omega}}(\tau) = {}^{b}\boldsymbol{\omega}(\tau) + \boldsymbol{b}_g(\tau) + \boldsymbol{n}_g(\tau) \end{cases} \tag{21}$$

其中 $\boldsymbol{e}_a(\tau)$ 和 $\boldsymbol{e}_g(\tau)$ 为高斯白噪声：

$$\boldsymbol{n}_a(\tau) = \sigma_a \cdot \frac{1}{\sqrt{\Delta\tau}} \cdot \boldsymbol{e}(\tau) \qquad \boldsymbol{n}_g(\tau) = \sigma_g \cdot \frac{1}{\sqrt{\Delta\tau}} \cdot \boldsymbol{e}(\tau) \qquad \boldsymbol{e}(\tau) \sim \mathcal{N}(\boldsymbol{0}_{3\times1}, \boldsymbol{I}_3) \tag{22}$$

且 σ_a 的单位为 $\left(\frac{m}{s^2} \times \frac{1}{\sqrt{hz}}\right)$，$\sigma_g$ 的单位为 $\left(\frac{rad}{s} \times \frac{1}{\sqrt{hz}}\right)$。而 $\boldsymbol{b}_a(\tau)$ 和 $\boldsymbol{b}_g(\tau)$ 为零偏，建模为随机游走：

$$\dot{\boldsymbol{b}}_a(\tau) = \sigma_{b_a} \cdot \boldsymbol{I}_3 \cdot \boldsymbol{e}(\tau) \qquad \dot{\boldsymbol{b}}_g(\tau) = \sigma_{b_g} \cdot \boldsymbol{I}_3 \cdot \boldsymbol{e}(\tau) \tag{23}$$

且 σ_{b_a} 的单位为 $\left(\frac{m}{s^3} \times \frac{1}{\sqrt{hz}}\right)$，$\sigma_{b_g}$ 的单位为 $\left(\frac{rad}{s^2} \times \frac{1}{\sqrt{hz}}\right)$，对其进行离散化后，有：

$$\boldsymbol{b}_a(\tau_{k+1}) = \boldsymbol{b}_a(\tau_k) + \sigma_{b_a} \cdot \sqrt{\Delta\tau} \cdot \boldsymbol{I}_3 \cdot \boldsymbol{e}(\tau) \tag{24}$$

注意，有 $\frac{1}{hz} = \Delta\tau$。

综上，对零偏连续时间状态方程进行离散化，得到：

$$\begin{pmatrix} \boldsymbol{b}_a(\tau_{k+1}) \\ \boldsymbol{b}_g(\tau_{k+1}) \end{pmatrix} = \begin{pmatrix} \boldsymbol{I}_3 & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_3 \end{pmatrix} \begin{pmatrix} \boldsymbol{b}_a(\tau_k) \\ \boldsymbol{b}_g(\tau_k) \end{pmatrix} + \sqrt{\Delta\tau} \cdot \begin{pmatrix} \sigma_{b_a} \cdot \boldsymbol{I}_3 & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \sigma_{b_a} \cdot \boldsymbol{I}_3 \end{pmatrix} \begin{pmatrix} \boldsymbol{e}(\tau_k) \\ \boldsymbol{e}(\tau_k) \end{pmatrix} \tag{25}$$

于是，有：

$$\boldsymbol{D}_{\boldsymbol{b}}(\tau_{k+1}) = \boldsymbol{D}_{\boldsymbol{b}}(\tau_k) + \Delta\tau \cdot \begin{pmatrix} \sigma_{b_a}^2 \cdot \boldsymbol{I}_3 & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \sigma_{b_g}^2 \cdot \boldsymbol{I}_3 \end{pmatrix} \tag{26}$$

使用的 **IMU** 为 **XSens MTI-G-710**，加速度计相关参数为：

$$\begin{cases} \sigma_a = 60\left(\frac{\mu g}{\sqrt{hz}}\right) = 60 \times 10^{-6} \times 9.8\left(\frac{m}{s^2} \times \frac{1}{\sqrt{hz}}\right) = 5.880 \times 10^{-4}\left(\frac{m}{s^2} \times \frac{1}{\sqrt{hz}}\right) \\ \sigma_{b_a} = \cdots \end{cases} \tag{27}$$

陀螺仪相关参数为:

$$\begin{cases} \sigma_g = 0.01 \left(\dfrac{deg}{s} \times \dfrac{1}{\sqrt{hz}} \right) = \dfrac{0.01 \times \pi}{180} \left(\dfrac{rad}{s} \times \dfrac{1}{\sqrt{hz}} \right) \approx 1.745 \times 10^{-4} \left(\dfrac{rad}{s} \times \dfrac{1}{\sqrt{hz}} \right) \\ \sigma_{b_g} = \cdots \end{cases} \tag{28}$$

5 基于卡尔曼滤波器的样条先验

5.1 零偏先验

以加速度计零偏为例，τ 时刻的零偏为：$\boldsymbol{b}_a(\tau)$。则基于速度随机游走模型，容易得到离散化后的状态方程为:

$$\boldsymbol{b}_a(\tau_{k+1}, \tau_k) = (\boldsymbol{I}_3) \cdot \boldsymbol{b}_a(\tau_k) + \left(\sigma_{b_a} \cdot \sqrt{\Delta\tau} \cdot \boldsymbol{I}_3 \right) \cdot \boldsymbol{e}(\tau_k) \tag{29}$$

其中 $\boldsymbol{b}_a(\tau_{k+1}, \tau_k)$ 为第 $k+1$ 时刻的预测状态，$\boldsymbol{b}_a(\tau_k)$ 为第 k 个时刻的状态，$\Delta\tau$ 为时间间隔。方差传播为:

$$\boldsymbol{D}_{b_a}(\tau_{k+1}, \tau_k) = \boldsymbol{D}_{b_a}(\tau_k) + \Delta\tau \cdot \sigma_{b_a}^2 \cdot \boldsymbol{I}_3 \tag{30}$$

基于得到的第 $k+1$ 个时刻的状态预测和方差信息，在增量优化的时候进行先验信息的约束。优化完成后，得到第 $k+1$ 时刻状态的虚拟量测值 $\hat{\boldsymbol{b}}_a(\tau_{k+1})$ 及其观测方差 $\hat{\boldsymbol{D}}(\tau_{k+1})$。而后进行更新操作:

$$\begin{aligned} \boldsymbol{b}_a(\tau_{k+1}) &= \boldsymbol{b}_a(\tau_{k+1}, \tau_k) + \boldsymbol{K}(\tau_{k+1}) \left(\hat{\boldsymbol{b}}_a(\tau_{k+1}) - \boldsymbol{b}_a(\tau_{k+1}, \tau_k) \right) \\ \boldsymbol{D}_{b_a}(\tau_{k+1}) &= (\boldsymbol{I}_3 - \boldsymbol{K}(\tau_{k+1})) \boldsymbol{D}_{b_a}(\tau_{k+1}, \tau_k) (\boldsymbol{I}_3 - \boldsymbol{K}(\tau_{k+1}))^\top + \boldsymbol{K}(\tau_{k+1}) \hat{\boldsymbol{D}}(\tau_{k+1}) \boldsymbol{K}^\top(\tau_{k+1}) \end{aligned} \tag{31}$$

其中:

$$\boldsymbol{K}(\tau_{k+1}) = \boldsymbol{D}_{b_a}(\tau_{k+1}, \tau_k) \left(\boldsymbol{D}_{b_a}(\tau_{k+1}, \tau_k) + \hat{\boldsymbol{D}}(\tau_{k+1}) \right)^{-1} \tag{32}$$

5.2 速度先验

6 边缘化

6.1 ceres 四元数求导相关问题

四元数对李代数求导:

$$\boldsymbol{q} = \begin{pmatrix} q_x & q_y & q_z & q_w \end{pmatrix}^\top = \begin{pmatrix} \dfrac{\boldsymbol{\theta}}{\theta} \sin \dfrac{\theta}{2} & \cos \dfrac{\theta}{2} \end{pmatrix} \tag{33}$$

其中:

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_x & \theta_y & \theta_z \end{pmatrix} \qquad \theta = \sqrt{(\theta_x)^2 + (\theta_y)^2 + (\theta_z)^2} \tag{34}$$

于是对于右扰动模型 (**VINS-Mono** 使用的，**Sophus** 使用的)，有:

$$\boldsymbol{q} = \boldsymbol{q} \circ \delta\boldsymbol{q} = \mathcal{L}(\boldsymbol{q}) \cdot \begin{bmatrix} \dfrac{1}{2} \cdot \delta\boldsymbol{\theta} \\ 1 \end{bmatrix} \qquad \frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\theta}} = \frac{1}{2} \cdot \mathcal{L}(\boldsymbol{q}) \cdot \begin{pmatrix} \boldsymbol{I}_3 & \boldsymbol{0}_3 \end{pmatrix}^\top = \frac{1}{2} \cdot \left(\begin{array}{ccc} q_w & -q_z & q_y \\ q_z & q_w & -q_x \\ -q_y & q_x & q_w \\ \hline -q_x & -q_y & -q_z \end{array} \right) \tag{35}$$

其中 $\mathcal{L}(\boldsymbol{q})$ 为 $\boldsymbol{q}$ 对应的左乘矩阵：

$$\mathcal{L}(\boldsymbol{q}) = \begin{pmatrix} q_w \cdot \boldsymbol{I}_3 + [\boldsymbol{q}_v]_\times & \boldsymbol{q}_v \\ -\boldsymbol{q}_v^\top & q_w \end{pmatrix} = \left(\begin{array}{ccc|c} q_w & -q_z & q_y & q_x \\ q_z & q_w & -q_x & q_y \\ -q_y & q_x & q_w & q_z \\ \hline -q_x & -q_y & -q_z & q_w \end{array}\right) \tag{36}$$

同样可以推到得到左扰动模型导数为 (Ceres 使用的)：

$$\boldsymbol{q} = \delta\boldsymbol{q} \circ \boldsymbol{q} = \mathcal{R}(\boldsymbol{q}) \cdot \begin{bmatrix} \frac{1}{2} \cdot \delta\boldsymbol{\theta} \\ 1 \end{bmatrix} \qquad \frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\theta}} = \frac{1}{2} \cdot \mathcal{R}(\boldsymbol{q}) \cdot \begin{pmatrix} \boldsymbol{I}_3 & \boldsymbol{0}_3 \end{pmatrix}^\top = \frac{1}{2} \cdot \left(\begin{array}{ccc} q_w & q_z & -q_y \\ -q_z & q_w & q_x \\ q_y & -q_x & q_w \\ \hline -q_x & -q_y & -q_z \end{array}\right) \tag{37}$$

6.2 ceres 评估函数

参数块的 **Global Size** 指的是其实际大小，比如用四元素表示的旋转有 **4** 个参数，三维系下的重力向量有 **3** 个参数。参数块的 **Local Size** 指的是其自由度，比如用四元素表示的旋转有 **3** 个自由度 (单位四元数模长为 **1**)，三维系下的重力向量有 **2** 个自由度 (重力向量模长为 **9.8**)。不在流形上进行优化的参数的 **Global Size** 和 **Local Size** 相同，比如位移状态向量。

参数快的尺寸直接决定了求解的雅克比矩阵的大小，**ceres** 提供了两者求解雅可比矩阵的方法：

Listing 1: Problem 类的提供的雅克比矩阵求解

```
bool Problem::Evaluate(const EvaluateOptions& options, double* cost,
                std::vector<double>* residuals, std::vector<double>* gradient,
                CRSMatrix* jacobian);
```

Key points:

1. **Evaluate Problem. Any of the output pointers can be nullptr. Which residual blocks and parameter blocks are used is controlled by the EvaluateOptions struct.**

Listing 2: EvaluateOptions

```
// Options struct to control Problem::Evaluate.
  struct EvaluateOptions {
    // The set of parameter blocks for which evaluation should be
    // performed. This vector determines the order that parameter blocks occur
    // in the gradient vector and in the columns of the jacobian matrix. If
    // parameter_blocks is empty, then it is assumed to be equal to vector
    // containing ALL the parameter blocks. Generally speaking the parameter
    // blocks will occur in the order in which they were added to the
    // problem. But, this may change if the user removes any parameter blocks
    // from the problem.
    //
    // NOTE: This vector should contain the same pointers as the ones used to
    // add parameter blocks to the Problem. These parameter block should NOT
    // point to new memory locations. Bad things will happen otherwise.
    std::vector<double*> parameter_blocks;

    // The set of residual blocks to evaluate. This vector determines the order
    // in which the residuals occur, and how the rows of the jacobian are
    // ordered. If residual_blocks is empty, then it is assumed to be equal to
    // the vector containing ALL the residual blocks. Generally speaking the
    // residual blocks will occur in the order in which they were added to the
    // problem. But, this may change if the user removes any residual blocks
    // from the problem.
    std::vector<ResidualBlockId> residual_blocks;

    // Even though the residual blocks in the problem may contain loss
    // functions, setting apply_loss_function to false will turn off the
    // application of the loss function to the output of the cost function. This
    // is of use for example if the user wishes to analyse the solution quality
    // by studying the distribution of residuals before and after the solve.
    bool apply_loss_function = true;

    int num_threads = 1;
  };
```

2. **The evaluation will use the values stored in the memory locations pointed to by the parameter block pointers used at the time of the construction of the problem.**

3. **If no LocalParameterizations or Manifolds are used, then the size of the gradient vector (and the number of columns in the jacobian) is the sum of the sizes of all the parameter blocks. If a parameter block has a LocalParameterization or Manifold, then it contributes "TangentSize" entries to the gradient vector (and the number of columns in the jacobian).**

Listing 3: CostFunction 类的提供的雅克比矩阵求解

```
virtual bool CostFunction::Evaluate(double const* const* parameters,
                                    double* residuals, double** jacobians) const = 0;
```

Key points:

1. **parameters has the same number of elements as parameter-block-sizes. Parameter blocks are in the same order as parameter-block-sizes.**

2. **A more interesting and common use is to impose constraints on the parameters. If the initial values of the parameter blocks satisfy the constraints, then returning**

false whenever the constraints are not satisfied will prevent the solver from moving into the infeasible region. This is not a very sophisticated mechanism for enforcing constraints, but is often good enough.

二者最主要的区别在于 **CostFunction** 类的提供的雅克比矩阵求解只考虑了参数块的 **Global Size**，而 **Problem** 类的提供的雅克比矩阵求解考虑参数块 **Local Size**。最主要的原因是 **Problem** 类知道参数块的流形优化信息，所以可以提供更进一步的雅克比矩阵。

Listing 4: 流形优化

```
    Problem::SetManifold(gravity, new ceres::SphereManifold<3>());
    Problem::SetManifold(SO3_BiToBc, new ceres::EigenQuaternionManifold());
```

与此类似的，当为某个残差函数 (cost function) 指定损失函数 (loss function) 的时候，**Problem** 类知道，但是 **CostFunction** 类不知道，所以恢复得到的残差向量中，**Problem** 类得到的包含了损失函数信息，所以如果是由 **CostFunction** 类计算得到的残差信息，需要通过损失函数映射残差。具体参考 **ceres** 官网 (`http://www.ceres-solver.org/nnls_modeling.html#lossfunction`)。

6.3 舒尔补

使用 **ceres** 优化，每次迭代优化求解线性方程组：

$$\left(\boldsymbol{J}^\top\boldsymbol{J}\right)\cdot\delta\boldsymbol{x}=\left(-\boldsymbol{J}^\top\boldsymbol{e}\right)\Rightarrow\boldsymbol{H}\cdot\delta\boldsymbol{x}=\boldsymbol{b}\tag{38}$$

现令 $\delta\boldsymbol{x}$ 由待边缘化的状态 $\delta\boldsymbol{x}_m$ 和不边缘化的状态 $\delta\boldsymbol{x}_n$ 构成，则有：

$$\begin{pmatrix}\boldsymbol{h}_{mm}&\boldsymbol{h}_{mn}\\\boldsymbol{h}_{nm}&\boldsymbol{h}_{nn}\end{pmatrix}\cdot\begin{pmatrix}\delta\boldsymbol{x}_m\\\delta\boldsymbol{x}_n\end{pmatrix}=\begin{pmatrix}\boldsymbol{b}_m\\\boldsymbol{b}_n\end{pmatrix}\tag{39}$$

对上式进行等价变换：

$$\left(\begin{array}{cc|c}\boldsymbol{h}_{mm}&\boldsymbol{h}_{mn}&\boldsymbol{b}_m\\\boldsymbol{h}_{nm}&\boldsymbol{h}_{nn}&\boldsymbol{b}_n\end{array}\right)\xrightarrow{r_2-\boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot r_1}\left(\begin{array}{cc|c}\boldsymbol{h}_{mm}&\boldsymbol{h}_{mn}&\boldsymbol{b}_m\\\boldsymbol{0}_{nm}&\boldsymbol{h}_{nn}-\boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot\boldsymbol{h}_{mn}&\boldsymbol{b}_n-\boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot\boldsymbol{b}_m\end{array}\right)\tag{40}$$

最终得到：

$$\left(\boldsymbol{h}_{nn}-\boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot\boldsymbol{h}_{mn}\right)\cdot\delta\boldsymbol{x}_n=\boldsymbol{b}_n-\boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot\boldsymbol{b}_m\Rightarrow\boldsymbol{H}_*\cdot\delta\boldsymbol{x}_n=\boldsymbol{b}_*\tag{41}$$

注意到，上式中的 $\boldsymbol{H}_*$ 和 $\boldsymbol{b}_*$ 是基于当前状态计算得到的。当 $\boldsymbol{x}$ 变化的时候，二者也会发生变化。为此，对 $\boldsymbol{b}^*$ 进行线性化操作：

$$\boldsymbol{b}^{(k+1)}=\boldsymbol{b}^{(k)}+\frac{\partial\boldsymbol{b}}{\partial\boldsymbol{x}}\cdot\delta\boldsymbol{x}=\boldsymbol{b}^{(k)}+\frac{\partial\boldsymbol{b}}{\partial e}\cdot\frac{\partial e}{\partial\boldsymbol{x}}\cdot\delta\boldsymbol{x}=\boldsymbol{b}^{(k)}-\boldsymbol{J}^\top\boldsymbol{J}\cdot\delta\boldsymbol{x}=\boldsymbol{b}^{(k)}-\boldsymbol{H}\cdot\delta\boldsymbol{x}\tag{42}$$

由此得到：

$$\begin{cases}\boldsymbol{b}_m^{(k+1)}=\boldsymbol{b}_m^{(k)}-\boldsymbol{h}_{mm}\cdot\delta\boldsymbol{x}_m-\boldsymbol{h}_{mn}\cdot\delta\boldsymbol{x}_n\\\boldsymbol{b}_n^{(k+1)}=\boldsymbol{b}_n^{(k)}-\boldsymbol{h}_{nm}\cdot\delta\boldsymbol{x}_m-\boldsymbol{h}_{nn}\cdot\delta\boldsymbol{x}_n\end{cases}\tag{43}$$

即有:

$$\begin{aligned}\boldsymbol{b}_*^{(k+1)} &= \boldsymbol{b}_n^{(k+1)} - \boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot\boldsymbol{b}_m^{(k+1)} \\ &= \left(\boldsymbol{b}_n^{(k)} - \boldsymbol{h}_{nm}\cdot\delta\boldsymbol{x}_m - \boldsymbol{h}_{nn}\cdot\delta\boldsymbol{x}_n\right) - \boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot\left(\boldsymbol{b}_m^{(k)} - \boldsymbol{h}_{mm}\cdot\delta\boldsymbol{x}_m - \boldsymbol{h}_{mn}\cdot\delta\boldsymbol{x}_n\right) \\ &= \left(\boldsymbol{b}_n^{(k)} - \boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot\boldsymbol{b}_m^{(k)}\right) - \left(\boldsymbol{h}_{nn} - \boldsymbol{h}_{nm}\cdot\boldsymbol{h}_{mm}^{-1}\cdot\boldsymbol{h}_{mn}\right)\cdot\delta\boldsymbol{x}_n \\ &= \boldsymbol{b}_*^{(k)} - \boldsymbol{H}_*\cdot\delta\boldsymbol{x}_n\end{aligned} \tag{44}$$

对上式进一步化简，得到:

$$\boldsymbol{b}_*^{(k+1)} = \boldsymbol{b}_*^{(k)} - \boldsymbol{H}_*\cdot\delta\boldsymbol{x}_n = \boldsymbol{J}_*^\top\cdot\left(\left(\boldsymbol{J}_*^\top\right)^{-1}\cdot\boldsymbol{b}_*^{(k)} - \boldsymbol{J}_*\cdot\delta\boldsymbol{x}_n\right) \tag{45}$$

所以残差为:

$$\boldsymbol{e}_* = -\left(\left(\boldsymbol{J}_*^\top\right)^{-1}\cdot\boldsymbol{b}_*^{(k)} - \boldsymbol{J}_*\cdot\delta\boldsymbol{x}_n\right) = \boldsymbol{J}_*\cdot\delta\boldsymbol{x}_n - \left(\boldsymbol{J}_*^\top\right)^{-1}\cdot\boldsymbol{b}_*^{(k)} \tag{46}$$

6.4 测试案例

Consider now a slightly more complicated example –the minimization of Powell’ s function. Let $\boldsymbol{x} = (x_1, x_2, x_3, x_4)^\top$ and

$$\begin{aligned}e_1(\boldsymbol{x}) &= x_1 + 10x_2 \\ e_2(\boldsymbol{x}) &= \sqrt{5}\,(x_3 - x_4) \\ e_3(\boldsymbol{x}) &= (x_2 - 2x_3)^2 \\ e_4(\boldsymbol{x}) &= \sqrt{10}\,(x_1 - x_4)^2\end{aligned} \tag{47}$$

容易得到:

$$\boldsymbol{J} = \begin{pmatrix} 1 & 10 & 0 & 0 \\ 0 & 0 & \sqrt{5} & -\sqrt{5} \\ 0 & 2\,(x_2 - 2x_3) & -4\,(x_2 - 2x_3) & 0 \\ 2\sqrt{10}\,(x_1 - x_4) & 0 & 0 & -2\sqrt{10}\,(x_1 - x_4)\end{pmatrix} \tag{48}$$

现在边缘化 x_2 和 x_4，重排参数向量为 $\boldsymbol{x} = (x_2, x_4 \,|\, x_1, x_3)^\top$，则有:

$$\boldsymbol{J} = \left(\begin{array}{cc|cc} 10 & 0 & 1 & 0 \\ 0 & -\sqrt{5} & 0 & \sqrt{5} \\ 2\,(x_2 - 2x_3) & 0 & 0 & -4\,(x_2 - 2x_3) \\ 0 & -2\sqrt{10}\,(x_1 - x_4) & 2\sqrt{10}\,(x_1 - x_4) & 0\end{array}\right) \tag{49}$$

当取 $\boldsymbol{x} = (x_2, x_4 \,|\, x_1, x_3)^\top = (-1, 1 \,|\, 3, 0)^\top$ 时:

$$\boldsymbol{J} = \left(\begin{array}{cc|cc} 10 & 0 & 1 & 0 \\ 0 & -\sqrt{5} & 0 & \sqrt{5} \\ -2 & 0 & 0 & 4 \\ 0 & -4\sqrt{10} & 4\sqrt{10} & 0\end{array}\right) \qquad \boldsymbol{H} = \boldsymbol{J}^\top\boldsymbol{J} = \left(\begin{array}{cc|cc} 104 & 0 & 10 & -8 \\ 0 & 165 & -160 & -5 \\ \hline 10 & -160 & 161 & 0 \\ -8 & -5 & 0 & 21\end{array}\right) \tag{50}$$

$$\boldsymbol{e}=\begin{pmatrix}-7 & -\sqrt{5} & 1 & 4\sqrt{10}\end{pmatrix}^\top \qquad \boldsymbol{b}=-\boldsymbol{J}^\top\cdot\boldsymbol{e}=\begin{pmatrix}72 & 155 & -153 & 1\end{pmatrix}^\top \tag{51}$$

$$\boldsymbol{J}_*=\begin{pmatrix}-1.90881 & -0.47586\\ 1.11507 & -4.47288\end{pmatrix} \qquad \left(\boldsymbol{J}_*^\top\right)^{-1}\cdot\boldsymbol{b}_*=\begin{pmatrix}3.36340\\ -2.86972\end{pmatrix} \tag{52}$$

7 速度可视化

对于世界坐标系 $\{w\}$ 下的某个点 ${}^w\boldsymbol{p}$，有：

$$ {}^w\boldsymbol{p}={}^w_b\boldsymbol{R}(\tau)\cdot{}^b\boldsymbol{p}(\tau)+{}^w\boldsymbol{p}_b(\tau) \tag{53}$$

对时间求导，有 (考虑世界系下的点是静止的)：

$$ {}^w\dot{\boldsymbol{p}}=-\left[{}^w_b\boldsymbol{R}(\tau)\cdot{}^b\boldsymbol{p}(\tau)\right]_\times\cdot{}^w_b\dot{\boldsymbol{R}}(\tau)+{}^w_b\boldsymbol{R}(\tau)\cdot{}^b\dot{\boldsymbol{p}}(\tau)+{}^w\dot{\boldsymbol{p}}_b(\tau)=\boldsymbol{0}_{3\times1} \tag{54}$$

得到该点相对于 $\{b\}$ 系的线速度在 $\{b\}$ 系下的表示：

$$ {}^b\dot{\boldsymbol{p}}(\tau)={}^w_b\boldsymbol{R}^\top(\tau)\cdot\left(\left[{}^w_b\boldsymbol{R}(\tau)\cdot{}^b\boldsymbol{p}(\tau)\right]_\times\cdot{}^w_b\dot{\boldsymbol{R}}(\tau)-{}^w\dot{\boldsymbol{p}}_b(\tau)\right) \tag{55}$$

注：事实上，只需要将速度绘制到 $\{w\}$ 系下就行，**OpenGL** 会通过相机投影转到相机坐标系下。

8 样条解析求导

以 **4** 阶样条为例，$\mathbb{R}^3$ 空间下的样条可以表示为：

$$\boldsymbol{p}(u)=\begin{pmatrix}\boldsymbol{p}_i & \boldsymbol{p}_{i+1} & \boldsymbol{p}_{i+2} & \boldsymbol{p}_{i+3}\end{pmatrix}\cdot\boldsymbol{M}^{(k)}\cdot\boldsymbol{u} \tag{56}$$

其中**:**

$$u=\frac{t-t_0}{\Delta t}-i \qquad \boldsymbol{u}=\begin{pmatrix}u^0 & u^1 & u^2 & u^3\end{pmatrix}^\top \qquad \boldsymbol{M}^{(k)}=\frac{1}{6}\times\begin{pmatrix}1 & -3 & 3 & -1\\ 4 & 0 & -6 & 3\\ 1 & 3 & 3 & -3\\ 0 & 0 & 0 & 1\end{pmatrix} \tag{57}$$

$\boldsymbol{M}^{(k)}$ 称之为 **blending matrix**。由于后面需要用到 $\boldsymbol{u}$ 对时间的导数，所以事先计算其系数 **Base Coefficients**：

$$\frac{\partial\boldsymbol{u}}{\partial u}=\begin{pmatrix}u^0 & u^1 & u^2 & u^3\\ 0 & u^0 & 2u^1 & 3u^2\\ 0 & 0 & 2u^0 & 6u^1\\ 0 & 0 & 0 & 6u^0\end{pmatrix}\Rightarrow\begin{pmatrix}1 & 1 & 1 & 1\\ 0 & 1 & 2 & 3\\ 0 & 0 & 2 & 6\\ 0 & 0 & 0 & 6\end{pmatrix} \tag{58}$$

当然，该样条也可以表示为以下的形式：

$$\boldsymbol{p}(u)=\begin{pmatrix}\boldsymbol{p}_i & \boldsymbol{p}_{i+1}-\boldsymbol{p}_i & \boldsymbol{p}_{i+2}-\boldsymbol{p}_{i+1} & \boldsymbol{p}_{i+3}-\boldsymbol{p}_{i+2}\end{pmatrix}\cdot\widetilde{\boldsymbol{M}}^{(k)}\cdot\boldsymbol{u} \tag{59}$$

此时，$\widetilde{\boldsymbol{M}}^{(k)}$ 为 **blending matrix** 的累加形式，即 $\widetilde{m}_{j,n}^{(k)} = \sum_{s=j}^{k-1} m_{s,n}^{(k)}$，有：

$$\widetilde{\boldsymbol{M}}^{(k)} = \frac{1}{6} \times \begin{pmatrix} 6 & 0 & 0 & 0 \\ 5 & 3 & -3 & 1 \\ 1 & 3 & 3 & -2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{60}$$

此时，$\widetilde{\boldsymbol{M}}^{(k)}$ 的第一行为第一基元向量，本文使用 **blending matrix**。

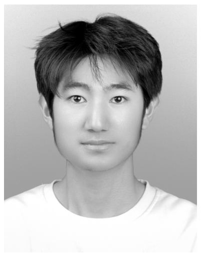
当计算样条的时间导数时，实际上是对 $\boldsymbol{u}$ 向量求时间导数。以一阶导为例：

$$\dot{\boldsymbol{p}}(t) = \frac{\partial \boldsymbol{p}(u)}{\partial t} = \begin{pmatrix} \boldsymbol{p}_i & \boldsymbol{p}_{i+1} & \boldsymbol{p}_{i+2} & \boldsymbol{p}_{i+3} \end{pmatrix} \cdot \boldsymbol{M}^{(k)} \cdot \frac{\partial \boldsymbol{u}}{\partial u} \cdot \frac{\partial u}{\partial t} \tag{61}$$

其中 $\dfrac{\partial \boldsymbol{u}}{\partial u}$ 直接可以基于之前计算得到的 **Base Coefficients** 获得，而 $\dfrac{\partial u}{\partial t} = \dfrac{1}{\Delta t}$。

9 ACKNOWLEDGMENT

Author Information



Shuolong Chen *received the B.S. degree in geodesy and geomatics engineering from Wuhan University, Wuhan China, in 2023. He is currently a master candidate at the school of Geodesy and Geomatics, Wuhan University. His area of research currently focuses on integrated navigation systems and multi-sensor fusion. Contact him via e-mail: shlchen@whu.edu.cn.*