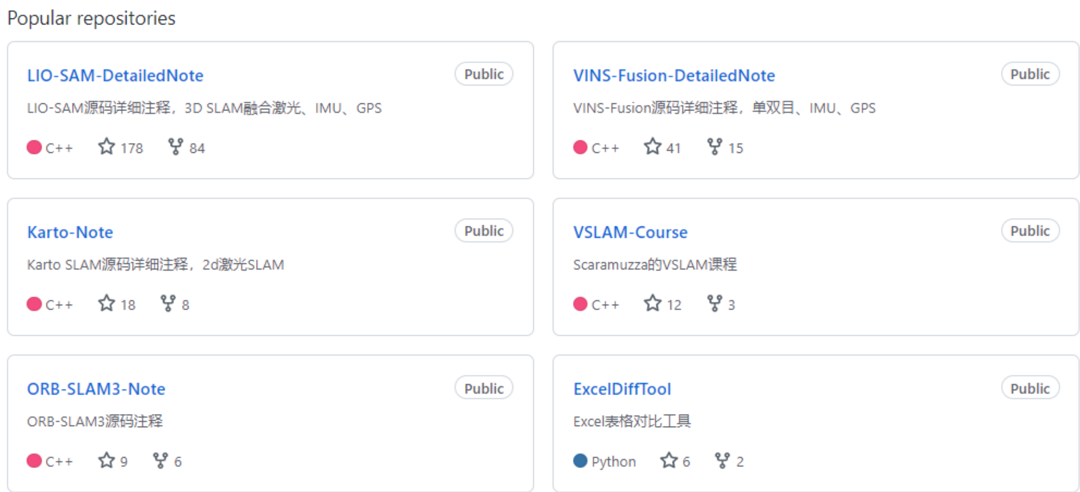
目录

各框架源码注释：https://github.com/smilefacehh



Vins:https://blog.csdn.net/huanghaihui_123/article/details/86518880?spm=1001.2101.3001.6650.11&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-11-86518880-blog-87357488.pc_relevant_antiscanv4&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-11-86518880-blog-87357488.pc_relevant_antiscanv4&utm_relevant_index=13
Vins 中相机和 IMU 对齐：https://zhuanlan.zhihu.com/p/466221991
Vins 边缘化：https://zhuanlan.zhihu.com/p/335242594
Vins 中 Ceres（BA）：https://zhuanlan.zhihu.com/p/488016175
vins 回环：https://blog.csdn.net/huanghaihui_123/article/details/87357488
Evo 工具使用：https://blog.51cto.com/u_14411234/3127894

Vins-Fusion

void sync_process()

While{
If（双目）
如果两目的图像时间间隔不超过 0.003 秒，则取出两帧图像 image0 和 image1。

inputImage（time , image0 , image1）

inputImageCnt++（输入图像数量加 1）
If (image1 为空)

trackImage（t , img）

1.如果先前已经有了预测的特征点（predict_pts），将 predict_pts 赋值给当前的特征点（cur_pts）

1.1 进行光流预测

cv::calcOpticalFlowPyrLK(prev_img, cur_img, prev_pts, cur_pts, status, ...)

如果返回的 status=1 的个数（succ_sum）小于 10，则增加金字塔层数继续进行预测 （根据 status 变量是否为 1 来确定对应的点是否被追踪到）。

如果之前并没有预测的特征点，则直接设置 3 层金字塔进行预测。

2.if（FLOW_BACK）

将当前图像与上一帧图像进行光流预测，将预测的点保存在 reverse_pts 中

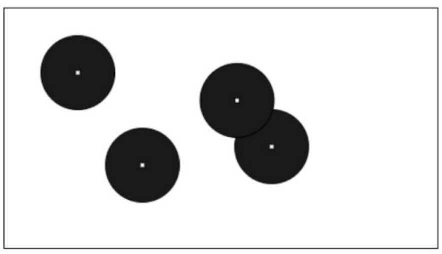
3. 去除外点（reduceVector），将丢失的点剔除

4 . setMask()

VINS-Fusion中特征提取与特征跟踪的实现

FeatureTracker::setMask函数：

- 将当前的特征点按照被连续追踪的次数从高到低排序
- 设置mask，使得角点提取的时候均匀分布



图：mask示意；从白色区域提取新的角点

5 .提取特征点

如果当前图像中特征点的数量少于配置文件中的数量（MAX_CNT），则进行特征点提取（goodFeatureToTrack），提取的特征点保存在 n_pts 中

提取成功后，将提取的点存入 cur_pts 中，id 存入 ids 中，将 1 存入 track_cnt 中

将 cur_pts 中的点投影到归一化平面中，并去除畸变（cur_un_pts）。并计算他的运动速度（pts_velocity）。

如果右目图像不为空，则执行上述相同的操作。

将当前帧的数据全部赋值为上一帧

```
prev_img = cur_img;
prev_pts = cur_pts;
prev_un_pts = cur_un_pts;
prev_un_pts_map = cur_un_pts_map;
prev_time = cur_time;
hasPrediction = false
```

6. 构建特征帧

将去除畸变后的归一化平面的点的 x,y,z（z=1）取出，将 cur_pts 中的像素坐标 x,y 赋值给 p_u,p_v，将速度 pts_velocity 取出赋值，最后得到 featureFrame

如果是双目，则对右目图像重复上述操作

最后返回 featureFrame

将 featureFrame 和时间 t 存入 featureBuf 中，如果系统不是多线程的话，开始 processMeasurements（）

如果是多线程的话，系统会在最开始读取配置文件参数，设置参数（setParameter）的时候就开始 processMeasurements（）

IMU 回调函数（imu_callback）

读取加速度和角速度 acc，gyr

InputIMU（acc，gyr）

① 将加速度和时间存入 accBuf 中，将角速度和时间存入 gyrBuf 中

②

fastPredictIMU（时间，加速度，角速度）

平均角速度： $\bar{\boldsymbol{\omega}}_i = \frac{1}{2}(\widehat{\boldsymbol{\omega}}_i + \widehat{\boldsymbol{\omega}}_{i+1}) - \boldsymbol{b}_{\omega_i}$

旋转量： $\widehat{\boldsymbol{\gamma}}_{i+1}^{b_k} = \widehat{\boldsymbol{\gamma}}_i^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}(\bar{\boldsymbol{\omega}}_i - \boldsymbol{b}_{\omega_i})\delta t \end{bmatrix}$

平均加速度： $\bar{\boldsymbol{a}}_i = \frac{1}{2}[\widehat{\boldsymbol{\gamma}}_i^{b_k}(\widehat{\boldsymbol{a}}_i - \boldsymbol{b}_{a_i}) + \widehat{\boldsymbol{\gamma}}_{i+1}^{b_k}(\widehat{\boldsymbol{a}}_{i+1} - \boldsymbol{b}_{a_i})]$

平移量： $\widehat{\boldsymbol{\alpha}}_{i+1}^{b_k} = \widehat{\boldsymbol{\alpha}}_i^{b_k} + \widehat{\boldsymbol{\beta}}_i^{b_k}\delta t + \frac{1}{2}\bar{\boldsymbol{a}}_i\delta t^2$

速度量： $\widehat{\boldsymbol{\beta}}_{i+1}^{b_k} = \widehat{\boldsymbol{\beta}}_i^{b_k} + \bar{\boldsymbol{a}}_i\delta t$

计算 IMU 预积分

latest_time（时间）、latest_Q（旋转量）、latest_P（平移量）、latest_V（速度量）、 latest_acc_0（上一帧的加速度）、latest_gyr_0（上一帧的角速度）。

特征回调（feature_callback）

构建 featureFrame

inputFeature（t , featureFrame）

将 featureFrame 和时间一起存入 featureBuf 中
如果不是多线程，则进行 processMeasurements（）

processMeasurements （）

如果 featureBuf 不为空，取出 featureBuf 中的第一帧图像 feature，将 feature 对应的时间+td（默认是 0）赋值给 curTime。
如果使用 IMU，则进行
getIMUInterval（前一帧图像对应的时间，当前帧对应时间，加速度（accVector），角速度（gryVector））
{这个函数主要是将在两帧图像之间的 IMU 的加速度和角速度分别保存在 accVector 和 gryVector 中，同时也会保存超过当前帧对应时间的后一帧 IMU 数据}

initFirstIMUPose（accVector）初始化第一帧 IMU 数据

{
https://blog.csdn.net/huanghaihui_123/article/details/103075107
计算 accVector 中所有加速度的平均值，将其转化成为旋转矩阵（R0）形式，在将 R0 转换成为欧拉角形式，取出偏航角 y。。。。。

}

processIMU

（当前 IMU 数据对应的时间（t），两个 IMU 之间的时间间隔（dt），加速度，角速度）

{

Push_back（dt，加速度，角速度）

{

将 dt，加速度，角速度存入 dt_buf，acc_buf，gry_buf 中

Propagate（dt，加速度，角速度）

{

midPointIntegration（）

{

① IMU 预积分

平均角速度：$$\overline{\boldsymbol{\omega}}_i = \frac{1}{2}(\widehat{\boldsymbol{\omega}}_i + \widehat{\boldsymbol{\omega}}_{i+1}) - \boldsymbol{b}_{\omega_i}$$

旋转量：$$\widehat{\boldsymbol{\gamma}}_{i+1}^{b_k} = \widehat{\boldsymbol{\gamma}}_i^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}(\overline{\boldsymbol{\omega}}_i - \boldsymbol{b}_{\omega_i})\delta t \end{bmatrix}$$

平均加速度：$$\overline{\boldsymbol{a}}_i = \frac{1}{2}[\widehat{\boldsymbol{\gamma}}_i^{b_k}(\widehat{\boldsymbol{a}}_i - \boldsymbol{b}_{a_i}) + \widehat{\boldsymbol{\gamma}}_{i+1}^{b_k}(\widehat{\boldsymbol{a}}_{i+1} - \boldsymbol{b}_{a_i})]$$

平移量：$$\widehat{\boldsymbol{\alpha}}_{i+1}^{b_k} = \widehat{\boldsymbol{\alpha}}_i^{b_k} + \widehat{\boldsymbol{\beta}}_i^{b_k}\delta t + \frac{1}{2}\overline{\boldsymbol{a}}_i \delta t^2$$

速度量：$$\widehat{\boldsymbol{\beta}}_{i+1}^{b_k} = \widehat{\boldsymbol{\beta}}_i^{b_k} + \overline{\boldsymbol{a}}_i \delta t$$

②计算雅克比矩阵（PPT 第五章-视觉惯性里程计(中)）

IMU误差传递模型离散化

$$\begin{bmatrix} \delta\boldsymbol{\alpha}_{k+1} \\ \delta\boldsymbol{\theta}_{k+1} \\ \delta\boldsymbol{\beta}_{k+1} \\ \delta\boldsymbol{b}_{a_{k+1}} \\ \delta\boldsymbol{b}_{w_{k+1}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & f_{01} & \delta t & f_{03} & f_{04} \\ 0 & f_{11} & 0 & 0 & -\delta t \\ 0 & f_{21} & \boldsymbol{I} & f_{23} & f_{24} \\ 0 & 0 & 0 & \boldsymbol{I} & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\alpha}_k \\ \delta\boldsymbol{\theta}_k \\ \delta\boldsymbol{\beta}_k \\ \delta\boldsymbol{b}_{a_k} \\ \delta\boldsymbol{b}_{w_k} \end{bmatrix} + \begin{bmatrix} v_{00} & v_{01} & v_{02} & v_{03} & 0 & 0 \\ 0 & \frac{-\delta t}{2} & 0 & \frac{-\delta t}{2} & 0 & 0 \\ -\frac{\boldsymbol{R}_k \delta t}{2} & v_{21} & -\frac{\boldsymbol{R}_{k+1}\delta t}{2} & v_{23} & 0 & 0 \\ 0 & 0 & 0 & 0 & \delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & \delta t \end{bmatrix} \begin{bmatrix} \boldsymbol{n}_{a_k} \\ \boldsymbol{n}_{w_k} \\ \boldsymbol{n}_{a_{k+1}} \\ \boldsymbol{n}_{w_{k+1}} \\ \boldsymbol{n}_{b_a} \\ \boldsymbol{n}_{b_w} \end{bmatrix}$$

$$f_{01} = \frac{\delta t}{2} f_{21} = -\frac{1}{4}\boldsymbol{R}_k(\widehat{\boldsymbol{a}}_k - \boldsymbol{b}_{a_k})^{\wedge}\delta t^2 - \frac{1}{4}\boldsymbol{R}_{k+1}(\widehat{\boldsymbol{a}}_{k+1} - \boldsymbol{b}_{a_k})^{\wedge}\left[I - \left(\frac{\widehat{\boldsymbol{\omega}}_k + \widehat{\boldsymbol{\omega}}_{k+1}}{2} - \boldsymbol{b}_{\omega_k}\right)^{\wedge}\delta t\right]\delta t^2$$

取自链接，里面有详细推导

$$f_{03} = -\frac{1}{4}(\boldsymbol{R}_k + \boldsymbol{R}_{k+1})\delta t^2$$

和代码对应；

（白噪声项正负号本质一样的）

$$f_{04} = \frac{\delta t}{2} f_{24} = \frac{1}{4}\boldsymbol{R}_{k+1}(\widehat{\boldsymbol{a}}_{k+1} - \boldsymbol{b}_{a_k})^{\wedge}\delta t^3$$

$$f_{11} = I - \left(\frac{\widehat{\boldsymbol{\omega}}_k + \widehat{\boldsymbol{\omega}}_{k+1}}{2} - \boldsymbol{b}_{\omega_k}\right)^{\wedge}\delta t$$

$$f_{21} = -\frac{1}{2}\boldsymbol{R}_k(\widehat{\boldsymbol{a}}_k - \boldsymbol{b}_{a_k})^{\wedge}\delta t - \frac{1}{2}\boldsymbol{R}_{k+1}(\widehat{\boldsymbol{a}}_{k+1} - \boldsymbol{b}_{a_k})^{\wedge}\left[I - \left(\frac{\widehat{\boldsymbol{\omega}}_k + \widehat{\boldsymbol{\omega}}_{k+1}}{2} - \boldsymbol{b}_{\omega_k}\right)^{\wedge}\delta t\right]\delta t$$

$$f_{23} = -\frac{1}{2}(\boldsymbol{R}_k + \boldsymbol{R}_{k+1})\delta t$$

$$f_{24} = \frac{1}{2}\boldsymbol{R}_{k+1}(\widehat{\boldsymbol{a}}_{k+1} - \boldsymbol{b}_{a_k})^{\wedge}\delta t^2$$

$$v_{00} = -\frac{1}{4}\boldsymbol{R}_k \delta t^2$$

$$v_{01} = v_{03} = \frac{\delta t}{2} v_{21} = \frac{1}{4}\boldsymbol{R}_{k+1}(\widehat{\boldsymbol{a}}_{k+1} - \boldsymbol{b}_{a_k})^{\wedge}\delta t^2 \frac{\delta t}{2}$$

$$v_{02} = -\frac{1}{4}\boldsymbol{R}_{k+1}\delta t^2$$

$$v_{21} = v_{23} = \frac{1}{4}\boldsymbol{R}_{k+1}(\widehat{\boldsymbol{a}}_{k+1} - \boldsymbol{b}_{a_k})^{\wedge}\delta t^2$$

31

}

将 midPointIntegration 得到的结果全部赋值进上一帧结果，dt 累加

}

}

}

processImage（图像，t）https://zhuanlan.zhihu.com/p/270382090

{

① addFeatureCheckParallax（）{根据设定的视差判断采取哪种边缘化策略}

② CalibrationExRotation（）计算相机和 IMU 之间的外参

{

1. solveRelativeR（通过 PnP 恢复出旋转矩阵，存入 Rc 中）
2. PPT 第五章-视觉惯性里程计（中）（二）P14-16

$$\left(\left[\mathbf{q}_{c_k}^{c_{k+1}}\right]_L - \left[\mathbf{q}_{b_k}^{b_{k+1}}\right]_R\right)\mathbf{q}_b^c = 0$$

PPT 第四章-惯性传感器部分 P31

$$\boldsymbol{q}_a\boldsymbol{q}_b = \begin{matrix} s_a s_b - x_a x_b - y_a y_b - z_a z_b + \\ (s_a x_b + x_a s_b + y_a z_b - z_a y_b)i + \\ (s_a y_b - x_a z_b + y_a s_b + z_a x_b)j + \\ (s_a z_b + x_a y_b - y_b x_a + z_a s_b)\ k \end{matrix} = \begin{bmatrix} s_b & z_b & -y_b & x_b \\ -z_b & s_b & x_b & y_b \\ y_b & -x_b & s_b & z_b \\ -x_b & -y_b & -z_b & s_b \end{bmatrix}\begin{bmatrix} x_a \\ y_a \\ z_a \\ s_a \end{bmatrix}$$

构建 L，R 矩阵，然后再进行 SVD 分解，求出 ric（相机到 IMU 之间的旋转（外参））

}

③ 初始化（单目+IMU）initialStructure（）

{

1. relativePose（）选择与最新帧有足够共视点，有足够的视差，并且能够解出 relative_R 和 relative_T 的帧 l
2. construct（）对 11 个关键帧做 sfm
3. 对滑动窗口内的所有帧做 sfm
4. VisualIMUAlignment 视觉 IMU 联合初始化，计算 bg、s、VS、g（视觉惯性对齐）

https://zhuanlan.zhihu.com/p/466221991

{

① solveGyroscopeBias（）初始化陀螺仪的 bias

PPT 第五章-视觉惯性里程计（中）（二）P18-19

$$\mathbf{q}_{b_{k+1}}^{c_0}{}^{-1} \otimes \mathbf{q}_{b_k}^{c_0} \otimes \hat{\boldsymbol{\gamma}}_{b_{k+1}}^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\mathbf{J}_{b_g}^{\gamma}\delta\mathbf{b}_g \end{bmatrix} = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ \frac{1}{2}\mathbf{J}_{b_g}^{\gamma}\delta\mathbf{b}_g \end{bmatrix} = \hat{\boldsymbol{\gamma}}_{b_{k+1}}^{b_k}{}^{-1} \otimes \mathbf{q}_{b_k}^{c_0}{}^{-1} \otimes \mathbf{q}_{b_{k+1}}^{c_0} \otimes \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}$$

$$\mathbf{J}_{b_g}^{\gamma}\delta\mathbf{b}_g = 2(\hat{\boldsymbol{\gamma}}_{b_{k+1}}^{b_k}{}^{-1} \otimes \mathbf{q}_{b_k}^{c_0}{}^{-1} \otimes \mathbf{q}_{b_{k+1}}^{c_0})_{vec}$$

$$\boldsymbol{x} = (\boldsymbol{A}^T\boldsymbol{A}).\mathrm{ldlt}().\mathrm{solve}(\boldsymbol{A}^T\boldsymbol{b})$$

② LinearAlignment（）PPT 第五章-视觉惯性里程计（中）（二）P21

- 首先是关于位置的预积分公式：

$$R_{c_0}^{b_k} P_{b_{k+1}}^{c_0} = R_{c_0}^{b_k}(P_{b_k}^{c_0} + v_{b_k}^{c_0}\Delta t_k - \frac{1}{2}g^{c_0}\Delta t_k{}^2) + \alpha$$

将 P 替换为含尺度信息 s 的式子，且 $v_{b_k}^{c_0} = R_{b_k}^{c_0} v_{b_k}$ ：

$$R_{c_0}^{b_k}(sP_{c_{k+1}}^{c_0} - R_{b_{k+1}}^{c_0} P_c^b) = R_{c_0}^{b_k}(sP_{c_k}^{c_0} - R_{b_k}^{c_0} P_c^b + R_{b_k}^{c_0} v_{b_k}\Delta t_k - \frac{1}{2}g^{c_0}\Delta t_k{}^2) + \alpha$$

观察上式，到目前为止，我们还未求出尺度信息 s 、图像帧的速度向量以及枢纽帧的重力向量。将系数中带有未知量的项挪到等号右侧，已知量挪到左侧，有：

$$\alpha - P_c^b + R_{c_0}^{b_k} R_{b_{k+1}}^{c_0} P_c^b = R_{c_0}^{b_k}(P_{c_{k+1}}^{c_0} - P_{c_k}^{c_0})s - v_{b_k}\Delta t_k + \frac{1}{2}R_{c_0}^{b_k} g^{c_0}\Delta t_k^2$$

- 关于速度的预积分公式：

$$R_{c_0}^{b_k} v_{b_{k+1}}^{c_0} = R_{c_0}^{b_k}(v_{b_k}^{c_0} - g^{c_0}\Delta t_k) + \beta_{b_{k+1}}^{b_k}$$

由于 $v_{b_{k+1}}^{c_0} = R_{b_{k+1}}^{c_0} v_{b_{k+1}}, v_{b_k}^{c_0} = R_{b_k}^{c_0} v_{b_k}$ ，将系数中含速度项和重力向量的未知量放到右侧，得：

$$\beta = -v_{b_k} + R_{c_0}^{b_k} R_{b_{k+1}}^{c_0} v_{b_{k+1}} + R_{c_0}^{b_k} g^{c_0}\Delta t_k$$

$$\mathbf{H}_{b_{k+1}}^{b_k} = \begin{bmatrix} -\mathbf{I}\Delta t_k & 0 & \frac{1}{2}\mathbf{R}_{c_0}^{b_k}\Delta t_k^2 & \mathbf{R}_{c_0}^{b_k}(\overline{\mathbf{p}}_{c_{k+1}}^{c_0} - \overline{\mathbf{p}}_{c_k}^{c_0}) \\ -\mathbf{I} & \mathbf{R}_{c_0}^{b_k}\mathbf{R}_{b_{k+1}}^{c_0} & \mathbf{R}_{c_0}^{b_k}\Delta t_k & 0 \end{bmatrix} \quad , \ \boldsymbol{\mathcal{X}}_I^k = \left[\mathbf{v}_{b_k}^{b_k}, \mathbf{v}_{b_{k+1}}^{b_{k+1}}, \mathbf{g}^{c_0}, s\right]$$

求解得到 s，g

优化重力向量 RefineGravity（）

再将位置预积分公式中的 g^{c_0} 替换为 $||g||\overrightarrow{g}+Bw$ ，同样将未知量移到等号右侧，则有：

$$\alpha-P_c^b+R_{c_0}^{b_k}R_{b_{k+1}}^{c_0}P_c^b-\frac{1}{2}R_{c_0}^{b_k}\Delta t_k^2||g||\overrightarrow{g}=R_{c_0}^{b_k}(P_{c_{k+1}}^{c_0}-P_{c_k}^{c_0})s-v_{b_k}\Delta t_k+\frac{1}{2}R_{c_0}^{b_k}\Delta t_k^2Bw$$

对位置的预积分做同样的操作，有：

$$\beta-R_{c_0}^{b_k}||g||\overrightarrow{g}\Delta t_k=-v_{b_k}+R_{c_0}^{b_k}R_{b_{k+1}}^{c_0}v_{b_{k+1}}+R_{c_0}^{b_k}Bw\Delta t_k$$

对应的 $Hx=b$ 变为：

$$\begin{bmatrix}-I\Delta t_k & 0 & \frac{1}{2}R_{c_0}^{b_k}\Delta t_k^2B & R_{c_0}^{b_k}(P_{c_{k+1}}^{c_0}-P_{c_k}^{c_0}) \\ -I & R_{c_0}^{b_k}R_{b_{k+1}}^{c_0} & R_{c_0}^{b_k}\Delta t_kB & 0\end{bmatrix}\begin{bmatrix}v_{b_k}\\ v_{b_{k+1}}\\ w\\ s\end{bmatrix}=\begin{bmatrix}\alpha_{b_{k+1}}^{b_k}-P_c^b+R_{c_0}^{b_k}R_{b_{k+1}}^{c_0}P_c^b-\frac{1}{2}R_{c_0}^{b_k}\Delta t_k^2||g||\overrightarrow{g}\\ \beta_{b_{k+1}}^{b_k}\end{bmatrix}$$

$$\mathbf{g}^{c_0}=\|\mathbf{g}\|\hat{\hat{\mathbf{g}}}^{c_0}+\boldsymbol{bw}$$

}

5. 更新状态，将初始化求出来的所有状态量对齐到第 0 帧 IMU 坐标系，同时要保证第 0 帧的 yaw=0。

之前求解的所有帧的位置信息均为没有尺度的 $T_c^w=[R_c^w|sP_c^w]$,现在我们乘上尺度信息并通过已知外参数 T_c^i 得到每一帧的 P_i^w 。已知：

$$T_i^w=T_c^w(T_c^i)^{-1}$$

展开得到：

$$\begin{bmatrix}R_i^w & P_i^w\\ 0 & 1\end{bmatrix}=\begin{bmatrix}R_c^w & sP_c^w\\ 0 & 1\end{bmatrix}\begin{bmatrix}R_c^{i\,T} & -R_c^{i\,T}P_c^i\\ 0 & 1\end{bmatrix}$$

得出：

$$P_i^w=sP_c^w-R_i^wP_c^i$$

```
/// 下面将所有状态量对齐到第0帧IMU坐标系
for (int i = frame_count; i >= 0; i--) // Ps[i]: 第0帧到第i帧的位移量
    Ps[i] = s * Ps[i] - Rs[i] * TIC[0] - (s * Ps[0] - Rs[0] * TIC[0]); // twi-tw0=ti0
```

```
Vs[kv] = frame_i->second.R * x.segment<3>(kv * 3); // Vw = Rwi*Vi ›
```

Vw = Rwi*Vi 将速度矢量由 IMU 系转到世界系

6. 重力对齐，最后再将滑窗里每一帧的位姿、速度矢量对齐到重力方向上，同时要保证第 0 帧的 yaw=0：

https://blog.csdn.net/huanghaihui_123/article/details/103075107

7. 三角化恢复深度

}

④ optimization（）

https://blog.csdn.net/weixin_39578197/article/details/110712642?spm=1001.2101.3001.6650.3&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-3-110712642-blog-121726485.pc_relevant_multi_platform_whitelistv1&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-3-110712642-blog-121726485.pc_relevant_multi_platform_whitelistv1&utm_relevant_index=6

https://www.jianshu.com/p/a9349370a8be/

1. 添加待优化变量[p,q](7)，[speed,ba,bg](9)，添加相机与 IMU 的外参[p_cb,q_cb](7)，添加时间偏移，添加边缘化的残差

2. 添加 IMU 的 residual。待优化变量分别为

优化变量：

$$[p_{b_k}^w, q_{b_k}^w],\ [v_{b_k}^w, b_{a_k}, b_{\omega_k}],\ [p_{b_{k+1}}^w, q_{b_{k+1}}^w],\ [v_{b_{k+1}}^w, b_{a_{k+1}}, b_{\omega_{k+1}}]$$

计算 Jacobian 时，残差对应的求偏导对象为上面的优化变量，但是计算时采用扰动方式计算，即扰动为$[\delta p_{b_k}^w, \delta\theta_{b_k}^w]$，$[\delta v_{b_k}^w, \delta b_{a_k}, \delta b_{\omega_k}]$，$[\delta p_{b_{k+1}}^w, \delta\theta_{b_{k+1}}^w]$，$[\delta v_{b_{k+1}}^w, \delta b_{a_{k+1}}, \delta b_{\omega_{k+1}}]$。

$$r_B{}^{15\times1}\left(\hat{z}_{b_{k+1}}^{b_k}, X\right) = \begin{bmatrix} \delta\alpha_{b_{k+1}}^{b_k} \\ \delta\theta_{b_{k+1}}^{b_k} \\ \delta\beta_{b_{k+1}}^{b_k} \\ \delta b_a \\ \delta b_g \end{bmatrix} = \begin{bmatrix} R_w^{b_k}\left(p_{b_{k+1}}^w - p_{b_k}^w - v_{b_k}^w \Delta t_k + \frac{1}{2} g^w \Delta t_k^2\right) - \alpha_{b_{k+1}}^{b_k} \\ 2\left[{\gamma_{b_{k+1}}^{b_k}}^{-1} \otimes {q_{b_k}^w}^{-1} \otimes q_{b_{k+1}}^w\right]_{xyz} \\ R_w^{b_k}\left(v_{b_{k+1}}^w - v_{b_k}^w + g^w \Delta t_k\right) - \beta_{b_{k+1}}^{b_k} \\ b_{a\,b_{k+1}} - b_{a\,b_k} \\ b_{\omega\,b_{k+1}} - b_{\omega\,b_k} \end{bmatrix}$$

直接求导　　李代数右扰动 $\frac{\partial(\boldsymbol{R}^{\top}\boldsymbol{p})}{\partial \boldsymbol{R}} = (\boldsymbol{R}^{\top}\boldsymbol{p})^{\wedge}$　　李代数右扰动 $\frac{\partial(\boldsymbol{R}^{\top}\boldsymbol{p})}{\partial \boldsymbol{R}} = (\boldsymbol{R}^{\top}\boldsymbol{p})^{\wedge}$

J[0]

$$J[0]^{15\times7} = \left[\frac{\partial r_B}{\partial p_{b_k}^w}, \frac{\partial r_B}{\partial q_{b_k}^w}\right] = \begin{bmatrix} -R_w^{b_k} & \left[R_w^{b_k}\left(p_{b_{k+1}}^w - p_{b_k}^w - v_{b_k}^w \Delta t_k + \frac{1}{2} g^w \Delta t_k^2\right)\right]^{\wedge} \\ 0 & -\mathcal{L}\left[{q_{b_{k+1}}^w}^{-1} \otimes q_{b_k}^w\right]\mathcal{R}\left[\gamma_{b_{k+1}}^{b_k}\right] \\ 0 & \left[R_w^{b_k}\left(v_{b_{k+1}}^w - v_{b_k}^w + g^w \Delta t_k\right)\right]^{\wedge} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

乘扰动

$$\begin{aligned} \frac{\partial \delta\theta_{b_{k+1}}^{b_k}}{\partial q_{b_k}^w} &= 2 \lim_{\delta\theta_{b_k}^w \to 0} \frac{{\gamma_{b_{k+1}}^{b_k}}^{-1} \otimes \left(q_{b_k}^w \otimes \begin{bmatrix} 1 \\ \frac{\delta\theta_{b_k}^w}{2} \end{bmatrix}\right)^{-1} \otimes q_{b_{k+1}}^w - {\gamma_{b_{k+1}}^{b_k}}^{-1} \otimes \left(q_{b_k}^w \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right)^{-1} \otimes q_{b_{k+1}}^w}{\delta\theta_{b_k}^w} \\ &= 2 \lim_{\delta\theta_{b_k}^w \to 0} \frac{{\gamma_{b_{k+1}}^{b_k}}^{-1} \otimes \begin{bmatrix} 1 \\ -\frac{\delta\theta_{b_k}^w}{2} \end{bmatrix} \otimes {q_{b_k}^w}^{-1} \otimes q_{b_{k+1}}^w - {\gamma_{b_{k+1}}^{b_k}}^{-1} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes {q_{b_k}^w}^{-1} \otimes q_{b_{k+1}}^w}{\delta\theta_{b_k}^w} \\ &= 2 \lim_{\delta\theta_{b_k}^w \to 0} \frac{\mathcal{R}\left[{q_{b_k}^w}^{-1} \otimes q_{b_{k+1}}^w\right]\mathcal{L}\left[{\gamma_{b_{k+1}}^{b_k}}^{-1}\right]\left(\begin{bmatrix} 1 \\ -\frac{\delta\theta_{b_k}^w}{2} \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right)}{\delta\theta_{b_k}^w} \end{aligned} \tag{A10}$$

$$
r_B{}^{15\times 1}\left(\hat{z}_{b_{k+1}}^{b_k}, X\right)=\begin{bmatrix}\delta\alpha_{b_{k+1}}^{b_k}\\ \delta\theta_{b_{k+1}}^{b_k}\\ \delta\beta_{b_{k+1}}^{b_k}\\ \delta b_a\\ \delta b_g\end{bmatrix}=\begin{bmatrix}R_w^{b_k}\left(p_{b_{k+1}}^w-p_{b_k}^w-v_{b_k}^w\Delta t_k+\frac{1}{2}g^w\Delta t_k^2\right)-\alpha_{b_{k+1}}^{b_k}\\ 2\left[{\gamma_{b_{k+1}}^{b_k}}^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w\right]_{xyz}\\ R_w^{b_k}\left(v_{b_{k+1}}^w-v_{b_k}^w+g^w\Delta t_k\right)-\beta_{b_{k+1}}^{b_k}\\ b_{a_{b_{k+1}}}-b_{a_{b_k}}\\ b_{\omega_{b_{k+1}}}-b_{\omega_{b_k}}\end{bmatrix}
$$

直接求导

PPT第五章-视觉惯性里程计(中)P31

PPT第五章-视觉惯性里程计(中)P31

J[1]

$$
J[1]^{15\times 9}=\left[\frac{\partial r_B}{\partial v_{b_k}^w},\frac{\partial r_B}{\partial b_{a_k}},\frac{\partial r_B}{\partial b_{w_k}}\right]=\begin{bmatrix}-R_w^{b_k}\Delta t & -J_{b_a}^{\alpha} & -J_{b_\omega}^{\alpha}\\ 0 & 0 & -\mathcal{L}\left[{q_{b_{k+1}}^w}^{-1}\otimes q_{b_k}^w\otimes \gamma_{b_{k+1}}^{b_k}\right]J_{b_\omega}^{\gamma}\\ -R_w^{b_k} & -J_{b_a}^{\beta} & -J_{b_\omega}^{\beta}\\ 0 & -I & 0\\ 0 & 0 & -I\end{bmatrix}
$$

$$
\frac{\partial\delta\theta_{b_{k+1}}^{b_k}}{\partial b_{\omega_k}}=2\lim_{\to 0}\frac{\left[\gamma_{b_{k+1}}^{b_k}\otimes\begin{bmatrix}1\\ \frac{1}{2}J_{b_\omega}^{\gamma}\delta b_\omega\end{bmatrix}\right]^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w-\left[\gamma_{b_{k+1}}^{b_k}\otimes\begin{bmatrix}1\\0\end{bmatrix}\right]^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w}{\delta b_{\omega_k}}
$$

$$
=2\lim_{\to 0}\frac{\begin{bmatrix}0\\ -\frac{1}{2}J_{b_\omega}^{\gamma}\delta b_\omega\end{bmatrix}\otimes {\gamma_{b_{k+1}}^{b_k}}^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w}{\delta b_{\omega_k}}
$$

$$
=-\mathcal{R}\left[{\gamma_{b_{k+1}}^{b_k}}^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w\right]\begin{bmatrix}0\\ J_{b_\omega}^{\gamma}\end{bmatrix}
$$

$$
=-\mathcal{L}\left[{q_{b_{k+1}}^w}^{-1}\otimes q_{b_k}^w\otimes \gamma_{b_{k+1}}^{b_k}\right]_{3\times 3}J_{b_\omega}^{\gamma}
$$

$$
r_B{}^{15\times 1}\left(\hat{z}_{b_{k+1}}^{b_k}, X\right)=\begin{bmatrix}\delta\alpha_{b_{k+1}}^{b_k}\\ \delta\theta_{b_{k+1}}^{b_k}\\ \delta\beta_{b_{k+1}}^{b_k}\\ \delta b_a\\ \delta b_g\end{bmatrix}=\begin{bmatrix}R_w^{b_k}\left(p_{b_{k+1}}^w-p_{b_k}^w-v_{b_k}^w\Delta t_k+\frac{1}{2}g^w\Delta t_k^2\right)-\alpha_{b_{k+1}}^{b_k}\\ 2\left[{\gamma_{b_{k+1}}^{b_k}}^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w\right]_{xyz}\\ R_w^{b_k}\left(v_{b_{k+1}}^w-v_{b_k}^w+g^w\Delta t_k\right)-\beta_{b_{k+1}}^{b_k}\\ b_{a_{b_{k+1}}}-b_{a_{b_k}}\\ b_{\omega_{b_{k+1}}}-b_{\omega_{b_k}}\end{bmatrix}
$$

J[2]

$$
J[2]^{15\times 7}=\left[\frac{\partial r_B}{\partial p_{b_{k+1}}^w},\frac{\partial r_B}{\partial q_{b_{k+1}}^w}\right]=\begin{bmatrix}R_w^{b_k} & 0\\ 0 & \mathcal{L}\left[{\gamma_{b_{k+1}}^{b_k}}^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w\right]\\ 0 & 0\\ 0 & 0\\ 0 & 0\end{bmatrix}
$$

$$
\frac{\partial\delta\theta_{b_{k+1}}^{b_k}}{\partial q_{b_{k+1}}^w}=2\lim_{\delta\theta_{b_k}^w\to 0}\frac{{\gamma_{b_{k+1}}^{b_k}}^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w\otimes\begin{bmatrix}1\\ \frac{\delta\theta_{b_{k+1}}^w}{2}\end{bmatrix}-{\gamma_{b_{k+1}}^{b_k}}^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w\otimes\begin{bmatrix}1\\0\end{bmatrix}}{\delta\theta_{b_{k+1}}^w}
$$

$$
=\mathcal{L}\left[{\gamma_{b_{k+1}}^{b_k}}^{-1}\otimes {q_{b_k}^w}^{-1}\otimes q_{b_{k+1}}^w\right]
$$

$$J[3]^{15\times 9}=\left[\frac{\partial r_B}{\partial v_{b_{k+1}}^{w}},\frac{\partial r_B}{\partial b_{a_{k+1}}},\frac{\partial r_B}{\partial b_{w_{k+1}}}\right]=\begin{bmatrix}0&0&0\\0&0&0\\R_w^{b_k}&0&0\\0&I&0\\0&0&I\end{bmatrix}$$

3. 添加视觉的 residual（以 ProjectionTwoFrameOneCamFactor 为例）

待估计量：

- 第i帧的$(\boldsymbol{t}_{wb_i},\boldsymbol{R}_{wb_i})$和第$j$帧的$(\boldsymbol{t}_{wb_j},\boldsymbol{R}_{wb_j})$（即每一帧的IMU在世界坐标系下的P和Q，这里用body系来表示，下标b）
- 该目与IMU的外参$(\boldsymbol{t}_{bc_1},\boldsymbol{R}_{bc_1})$
- 该特征点的逆深度λ
- 时间偏移（时间偏移的雅可比矩阵推导留作作业）

残差项：

$$\boldsymbol{r}_{ij}=\begin{bmatrix}\frac{X_{c_j}}{Z_{c_j}}-u_{c_j}\\ \frac{Y_{c_j}}{Z_{c_j}}-v_{c_j}\end{bmatrix}\quad\Rightarrow\quad \boldsymbol{J}(\boldsymbol{r}_{ij})=\begin{bmatrix}\frac{\partial \boldsymbol{r}_{ij}}{\partial\begin{bmatrix}\boldsymbol{t}_{wb_i}\\ \boldsymbol{R}_{wb_i}\end{bmatrix}} & \frac{\partial \boldsymbol{r}_{ij}}{\partial\begin{bmatrix}\boldsymbol{t}_{wb_j}\\ \boldsymbol{R}_{wb_j}\end{bmatrix}} & \frac{\partial \boldsymbol{r}_{ij}}{\partial\begin{bmatrix}\boldsymbol{t}_{bc_1}\\ \boldsymbol{R}_{bc_1}\end{bmatrix}} & \frac{\partial \boldsymbol{r}_{ij}}{\partial\lambda}\end{bmatrix}\quad ?$$

链式法则

$$\boxed{\frac{\partial \boldsymbol{r}_{ij}}{\partial \boldsymbol{P}_{c_j}}}\boldsymbol{J}\left(\boldsymbol{P}_{c_j}\right),\qquad \boldsymbol{J}\left(\boldsymbol{P}_{c_j}\right)=\begin{bmatrix}\frac{\partial \boldsymbol{P}_{c_j}}{\partial\begin{bmatrix}\boldsymbol{t}_{wb_i}\\ \boldsymbol{R}_{wb_i}\end{bmatrix}} & \frac{\partial \boldsymbol{P}_{c_j}}{\partial\begin{bmatrix}\boldsymbol{t}_{wb_j}\\ \boldsymbol{R}_{wb_j}\end{bmatrix}} & \frac{\partial \boldsymbol{P}_{c_j}}{\partial\begin{bmatrix}\boldsymbol{t}_{bc_1}\\ \boldsymbol{R}_{bc_1}\end{bmatrix}} & \frac{\partial \boldsymbol{P}_{c_j}}{\partial\lambda}\end{bmatrix}$$

接下来逐项推导

$$\frac{\partial \boldsymbol{r}_{ij}}{\partial \boldsymbol{P}_{c_j}}=\begin{bmatrix}\frac{1}{Z_{c_j}} & 0 & -\frac{X_{c_j}}{Z_{c_j}^2}\\ 0 & \frac{1}{Z_{c_j}} & -\frac{Y_{c_j}}{Z_{c_j}^2}\end{bmatrix}$$

推导可以参考SLAM十四讲中BA推导的部分

公式（27）中 P 在第 i 个相机的像素坐标系下坐标为：

$$P_{uv_i} = \lambda_l \pi_c \left({T_{b\leftarrow c}}^{-1} {T_{w\leftarrow b_i}}^{-1} P_{w_l} \right)$$

$$\Leftrightarrow P_{w_l} = T_{w\leftarrow b_i} T_{b\leftarrow c} \frac{1}{\lambda_l} {\pi_c}^{-1} \left(P_{uv_i} \right)$$

$$P_{w_l} = R_{b_i}^{w} \left(R_c^b \frac{1}{\lambda_l} {\pi_c}^{-1} \left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix} \right) + p_c^b \right) + p_{b_i}^w$$

P 在第 j 个相机的相机坐标系下坐标为：

$$P_l^{c_j} = {T_{b\leftarrow c}}^{-1} {T_{w\leftarrow b_j}}^{-1} P_{w_l}$$

$$\Leftrightarrow P_{w_l} = T_{w\leftarrow b_j} T_{b\leftarrow c} P_l^{c_j}$$

$$P_{w_l} = R_{b_j}^{w} \left(R_c^b P_l^{c_j} + p_c^b \right) + p_{b_j}^w$$

将(A12)代入(A13)可得：

$$R_{b_i}^{w} \left(R_c^b \frac{1}{\lambda_l} {\pi_c}^{-1} \left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix} \right) + p_c^b \right) + p_{b_i}^w = R_{b_j}^{w} \left(R_c^b P_l^{c_j} + p_c^b \right) + p_{b_j}^w$$

$$\Rightarrow P_l^{c_j} = R_b^c \left\{ R_w^{b_j} \left[R_{b_i}^{w} \left(R_c^b \frac{1}{\lambda_l} {\pi_c}^{-1} \left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix} \right) + p_c^b \right) + p_{b_i}^w - p_{b_j}^w \right] - p_c^b \right\}$$

$$\boxed{= R_b^c \left\{ R_w^{b_j} \left[R_{b_i}^{w} \left(R_c^b \frac{1}{\lambda_l} \bar{P}_l^{c_i} + p_c^b \right) + p_{b_i}^w - p_{b_j}^w \right] - p_c^b \right\}}$$

将$\boldsymbol{P}_{c_j}$展开：

$$\boxed{\boldsymbol{P}_{c_j} = \mathbf{R}_{bc_1}^{\top} \mathbf{R}_{wb_j}^{\top} \mathbf{R}_{wb_i} \mathbf{R}_{bc_1} \boldsymbol{P}_{c_i} + \mathbf{R}_{bc_1}^{\top} \left(\mathbf{R}_{wb_j}^{\top} \left(\left(\mathbf{R}_{wb_i} \boldsymbol{t}_{bc_1} + \boldsymbol{t}_{wb_i} \right) - \boldsymbol{t}_{wb_j} \right) - \boldsymbol{t}_{bc_1} \right)}$$

J[0]

对 $\dfrac{\partial P_{c_j}}{\partial \begin{bmatrix} t_{wb_i} \\ R_{wb_i} \end{bmatrix}}$

- 对t_{wb_i}，$\dfrac{\partial P_{c_j}}{\partial t_{wb_i}} = \mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}$，（$t_{wb_i}$ 项的因子）
- 对R_{wb_i}，$\dfrac{\partial P_{c_j}}{\partial R_{wb_i}} = \dfrac{\partial \mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}R_{bc_1}P_{c_i} + \mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}t_{bc_1} + (\dots)}{\partial R_{wb_i}} = \dfrac{\partial \mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}(R_{bc_1}P_{c_i} + t_{bc_1})}{\partial R_{wb_i}} = \dfrac{\partial \mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}P_{b_i}}{\partial R_{wb_i}}$

对P_{b_i}右扰动，参考第10页ppt，得出：

$$\frac{\partial P_{c_j}}{\partial R_{wb_i}} = \frac{\partial \mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}P_{b_i}}{\partial R_{wb_i}} = -\mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}P_{b_i}{}^{\wedge}$$

J[1]

$$R_{b_i}^{w}\left(R_c^b \frac{1}{\lambda_l}\pi_c^{-1}\left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix}\right) + p_c^b\right) + p_{b_i}^w = R_{b_j}^w\left(R_c^b P_l^{c_j} + p_c^b\right) + p_{b_j}^w$$

$$\Rightarrow P_l^{c_j} = R_b^c\left\{R_w^{b_j}\left[R_{b_i}^w\left(R_c^b \frac{1}{\lambda_l}\pi_c^{-1}\left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix}\right) + p_c^b\right) + p_{b_i}^w - p_{b_j}^w\right] - p_c^b\right\}$$

$$= R_b^c\left\{ \boxed{R_w^{b_j}\left[\boxed{R_{b_i}^w \boxed{\left(R_c^b \frac{1}{\lambda_l}\bar{P}_l^{c_i} + p_c^b\right)} + p_{b_i}^w} - p_{b_j}^w\right]} - p_c^b\right\}$$

（第i帧某一特征点在世界坐标系下的坐标，对应pts_w）

Pbi
（第i帧某一特征点在IMU坐标系下的坐标，对应pts_imu_i）

Pbj
（第i帧某一特征点出现在第j帧时，IMU坐标系下的坐标，对应pts_imu_j）

$$\frac{\partial P_{cj}}{\partial t_{wbj}} = -R_b^c R_w^{bj}$$

$$\frac{\partial P_{cj}}{\partial R_{wbj}} = \frac{\partial R_b^c\left\{R_w^{bj}\left[R_{bi}^w\left(R_c^b \frac{1}{\lambda_l}\overline{P}_l^{ci} + p_{bi}^w\right) - p_{bj}^w\right] - p_c^b\right\}}{\partial R_{wbj}} = \frac{\partial R_b^c\left\{R_w^{bj}\left[p_{bi}^w - p_{bj}^w\right] - p_c^b\right\}}{\partial R_{wbj}} = R_b^c\left[R_w^{bj}P_{bj}\right]^{\wedge}$$

J[2]

对 $\dfrac{\partial \boldsymbol{P}_{c_j}}{\partial \begin{bmatrix} \boldsymbol{t}_{bc_1} \\ \boldsymbol{R}_{bc_1} \end{bmatrix}}$

- 对$\boldsymbol{t}_{bc_1}$，$\dfrac{\partial \boldsymbol{P}_{c_j}}{\partial \boldsymbol{t}_{bc_1}} = \mathbf{R}_{bc_1}^{\top}(\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i} - \boldsymbol{I})$
- 对$\boldsymbol{R}_{bc_1}$，$\dfrac{\partial \boldsymbol{P}_{c_j}}{\partial \boldsymbol{R}_{bc_1}} = \dfrac{\partial \mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}\boldsymbol{R}_{bc_1}\boldsymbol{P}_{c_i}}{\partial \boldsymbol{R}_{bc_1}} + \dfrac{\mathbf{R}_{bc_1}^{\top}\left(\mathbf{R}_{wb_j}^{\top}\left(\left(\mathbf{R}_{wb_i}\boldsymbol{t}_{bc_1}+\boldsymbol{t}_{wb_i}\right)-\boldsymbol{t}_{wb_j}\right)-\boldsymbol{t}_{bc_1}\right)}{\partial \boldsymbol{R}_{bc_1}}$

均添加右扰动，参考第10、11页ppt，得出：

$$\frac{\partial \boldsymbol{P}_{c_j}}{\partial \boldsymbol{R}_{bc_1}} = -\mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}\boldsymbol{R}_{bc_1}\boldsymbol{P}_{c_i}{}^{\wedge} + \left(\mathbf{R}_{bc_1}^{\top}\mathbf{R}_{wb_j}^{\top}\mathbf{R}_{wb_i}\boldsymbol{R}_{bc_1}\boldsymbol{P}_{c_i}\right)^{\wedge} + \left(\mathbf{R}_{bc_1}^{\top}\left(\mathbf{R}_{wb_j}^{\top}\left(\left(\mathbf{R}_{wb_i}\boldsymbol{t}_{bc_1}+\boldsymbol{t}_{wb_i}\right)-\boldsymbol{t}_{wb_j}\right)-\boldsymbol{t}_{bc_1}\right)\right)^{\wedge}$$

J[3]

$$R_{b_i}^{w}\left(R_c^b\frac{1}{\lambda_l}\pi_c{}^{-1}\left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix}\right)+p_c^b\right)+p_{b_i}^w = R_{b_j}^w\left(R_c^b P_l^{c_j}+p_c^b\right)+p_{b_j}^w$$

$$\Rightarrow P_l^{c_j} = R_b^c\left\{R_w^{b_j}\left[R_{b_i}^w\left(R_c^b\frac{1}{\lambda_l}\pi_c{}^{-1}\left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix}\right)+p_c^b\right)+p_{b_i}^w-p_{b_j}^w\right]-p_c^b\right\}$$

$$= R_b^c\left\{R_w^{b_j}\left[R_{b_i}^w\left(R_c^b\frac{1}{\lambda_l}\bar{P}_l^{c_i}+p_c^b\right)+p_{b_i}^w-p_{b_j}^w\right]-p_c^b\right\}$$

$$\frac{\partial P_{cj}}{\partial \lambda} = \frac{\partial R_b^c\left\{R_w^{bj}\left[R_{bi}^w\left(R_c^b\dfrac{1}{\lambda_l}\overline{P}_l^{ci}+p_{bi}^w\right)-p_{bj}^w\right]-p_c^b\right\}}{\partial \lambda} = -\frac{R_b^c R_w^{bj} R_{bi}^w R_c^b \overline{P}_l^{ci}}{\lambda^2}$$

J[4]

$$R^w_{b_i}\left(R^b_c \frac{1}{\lambda_l}\pi_c{}^{-1}\left(\begin{bmatrix}u_l^{c_i}\\ v_l^{c_i}\end{bmatrix}\right)+p^b_c\right)+p^w_{b_i} = R^w_{b_j}\left(R^b_c P_l^{c_j}+p^b_c\right)+p^w_{b_j}$$

$$\Rightarrow P_l^{c_j} = R^c_b\left\{R^{b_j}_w\left[R^w_{b_i}\left(R^b_c\frac{1}{\lambda_l}\pi_c{}^{-1}\left(\begin{bmatrix}u_l^{c_i}\\ v_l^{c_i}\end{bmatrix}\right)+p^b_c\right)+p^w_{b_i}-p^w_{b_j}\right]-p^b_c\right\}$$

$$= R^c_b\left\{R^{b_j}_w\left[R^w_{b_i}\left(R^b_c\frac{1}{\lambda_l}\bar{P}_l^{c_i}+p^b_c\right)+p^w_{b_i}-p^w_{b_j}\right]-p^b_c\right\}$$

$$\overline{P}_l^{ci} = P_{ci} - (td - tdi)\times v_i$$

$$\frac{\partial P_{cj}}{\partial td} = \frac{\partial R_b^c\left\{R_w^{bj}\left[R_{bi}^w\left(R_c^b\frac{1}{\lambda_l}\overline{P}_l^{ci}+p_{bi}^w\right)-p_{bj}^w\right]-p_c^b\right\}}{\partial td} = -\frac{R_b^c R_w^{bj} R_{bi}^w R_c^b v_i}{\lambda}$$

4. 添加完成待优化变量后开始进行优化

5. 开始构建边缘化残差项（边缘化最老帧），即将需要边缘化的 factor 放入 marginalization_info，并指出需要边缘化掉的变量：para_Pose[0]、para_SpeedBias[0]以及窗口内第零帧第一次观测到的特征点 para_Feature[feature_index]。

https://zhuanlan.zhihu.com/p/335242594

https://www.freesion.com/article/42241448279/

https://blog.csdn.net/HozenChe/article/details/125291285

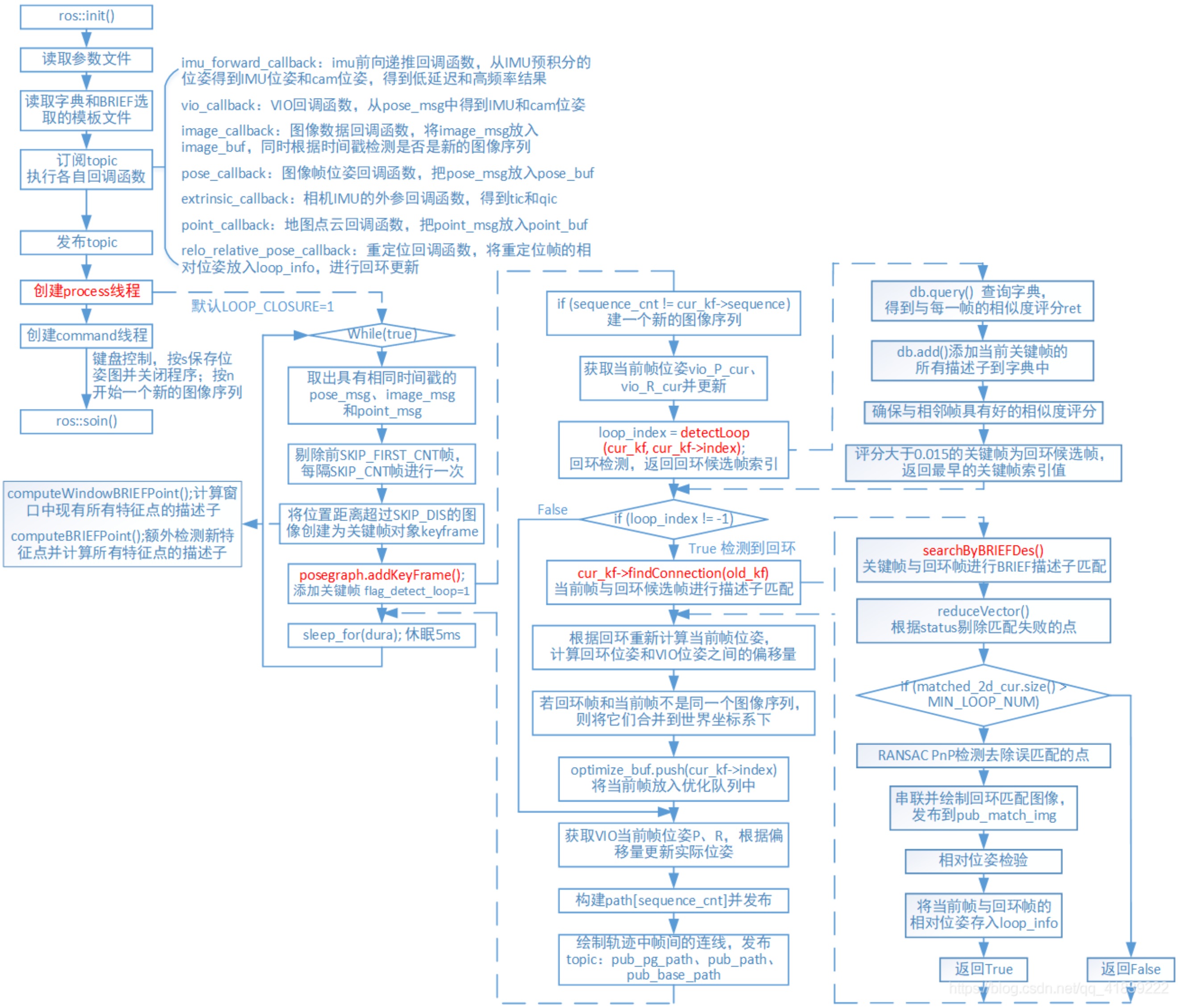
⑤ 更新状态

⑥ 滑窗 slideWindow（）

}

回环优化

https://blog.csdn.net/xiaojinger_123/article/details/119597747?utm_medium=distribute.pc_relevant.none-task-blog-2~default~baidujs_baidulandingword~default-0-119597747-blog-87357488.pc_relevant_blogantidownloadv1&spm=1001.2101.3001.4242.1&utm_relevant_index=3



全局优化

https://blog.csdn.net/hltt3838/article/details/109725845?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522165629167416782246472435%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fblog.%2522%257D&request_id=165629167416782246472435&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~blog~first_rank_ecpm_v1~rank_v31_ecpm-13-109725845-null-null.nonecase&utm_term=vins&spm=1018.2226.3001.4450