VINS-Mon:A Robust and Versatile Monocular Visual-INertial State Estimator

摘要

组成：单目加IMU

问题：IMU处理，评估器初始化，外参标定，非线性优化。

方法：

- 我的方法从一个可以完成评估器初始化和失败恢复的robust程序开始。
- 一个紧耦合的非线性优化方法杯用来获取一个高精度的VI里程计，通过融合pre-integrated IMU数据和visual数据。
- 一个回环检测模块通过结合紧耦合公式，能够实现一个最小的开销实现重定位。
- 额外的使用了4个自由度的pose graph来优化，强制它保持全局一致。

OVERVIEW

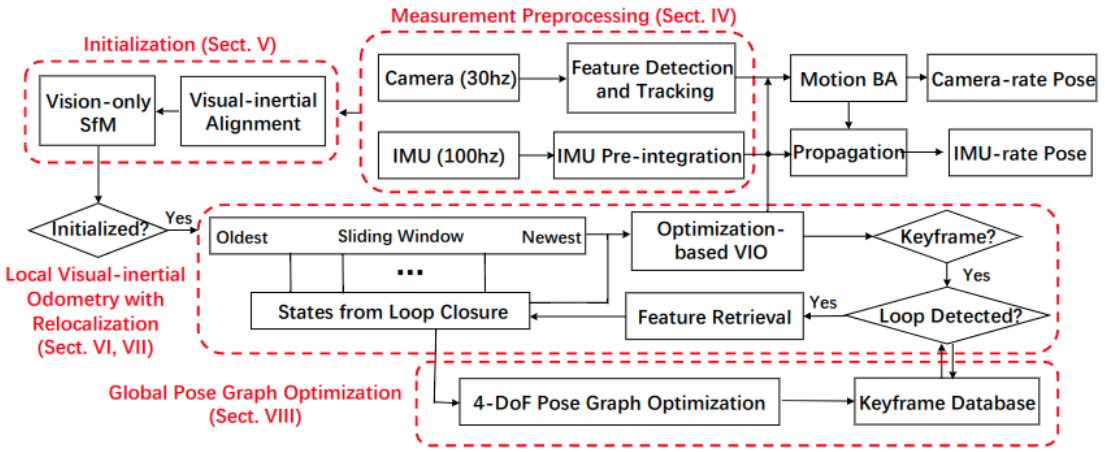


Fig. 2. A block diagram illustrating the full pipeline of the proposed monocular visual-inertial state estimator.

1. 从Measurement Preprocessing 模块启动
    - camera：完成Feature detection and Tracking
    - imu：连续两帧的测量值进行pre-integrated
2. Initialization
    - 提供用来引导subsequent nonlinear optimization-based VIO的所有必要的值，包括：位姿向量，速度向量，重力向量，陀螺仪偏差，3d 特征点坐标。
3. VIO with relocalization
    - 紧耦合了pre-integrated IMU measurements和feature observations
    - 回环检测
4. pose graph optimization
    - 在几何上确认重定位的结果
    - 执行global optimization 来减小漂移。

VIO,relocalization,and pose graph optimization 同时跑在多个线程上。每个模块有不同的运行速率和实时满足所有操作。

IV. MEASUREMENT PREPROCESSING

这个部分处理inertial and monocualr visual measurements。

A. Vision Processing Front-end

- 使用KLT sparse optical flow algorithm。
- 维持100-300个特征点数量
- 通过设置最小间距实现特征点的均匀分布。
- 2d 特征点先矫正，然后再通过了outlier rejection 后，将他们映射到一个单位球上。outlier rejection 通过使用对fundamental matrix model 模型使用RANSAC来实现。
- 选取关键帧，主要有两种方法
    - 通过两帧的视差来选取，如果视差大过一定的值的时候，选取为关键帧。但是由于旋转也会造成视差，而纯旋转的时候是无法完成triangualted的，所以为了避免这种情况，我们使用陀螺仪短期的积分来计算旋转，（然后再计算视差的时候减去旋转造成的部分？）。这个短期积分只是用来选取关键帧的时候使用，而不在其他地方使用，所以不会对其他地方造成影响。
    - 如果特征点的数量太少了的话，也设置为关键帧，这是为了避免丢失feature track。

B. IMU Pre-integration

> 写在前面，其实和forster那篇在流形上的预积分思路差不多，只不过这个是在四元数上做的。

$$\begin{aligned}\hat{\mathbf{a}}_t &= \mathbf{a}_t + \mathbf{b}_{a_t} + \mathbf{R}_w^t \mathbf{g}^w + \mathbf{n}_a \\ \hat{\boldsymbol{\omega}}_t &= \boldsymbol{\omega}_t + \mathbf{b}_{w_t} + \mathbf{n}_w .\end{aligned} \tag{1}$$

$\hat{a}_t$、$\hat{w}_t$是IMU的度数，度数的组成模型由式（1）表示，加速度包含了重力加速度、偏差、噪音的影响。角加速度包含了偏差和噪音的影响。

bias $\mathbf{b}_w$, and additive noise. We assume that the additive noise in acceleration and gyroscope measurements are Gaussian, $\mathbf{n}_a \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}_a^2)$, $\mathbf{n}_w \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}_w^2)$. Acceleration bias and gyroscope bias are modeled as random walk, whose derivatives are Gaussian, $\mathbf{n}_{b_a} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}_{b_a}^2)$, $\mathbf{n}_{b_w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}_{b_w}^2)$:

$$\dot{\mathbf{b}}_{a_t} = \mathbf{n}_{b_a}, \quad \dot{\mathbf{b}}_{w_t} = \mathbf{n}_{b_w}. \tag{2}$$

> 噪声符合高斯分布，random walk的每一步也符合高斯分布。
>
> random walk表示偏差会随着时间的变化而变化，同时，变化方向是无法预测的，并且跟过去状态相关。

从b_k到b_{k+1}帧的位置，速度和方向状态可以通过在这个时间$[t_k, t_{k+1}]$里面IMU测量数据来得到他们的传播。

$$\Delta t = t_{k+1} - t_k$$

$$
\begin{aligned}
\mathbf{p}_{b_{k+1}}^{w} &= \mathbf{p}_{b_k}^{w} + \mathbf{v}_{b_k}^{w}\Delta t_k \\
&\quad + \iint_{t\in[t_k,t_{k+1}]} \left(\mathbf{R}_t^w(\hat{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a) - \mathbf{g}^w\right) dt^2 \\
\mathbf{v}_{b_{k+1}}^{w} &= \mathbf{v}_{b_k}^{w} + \int_{t\in[t_k,t_{k+1}]} \left(\mathbf{R}_t^w(\hat{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a) - \mathbf{g}^w\right) dt \\
\mathbf{q}_{b_{k+1}}^{w} &= \mathbf{q}_{b_k}^{w} \otimes \int_{t\in[t_k,t_{k+1}]} \frac{1}{2}\boldsymbol{\Omega}(\hat{\boldsymbol{\omega}}_t - \mathbf{b}_{w_t} - \mathbf{n}_w)\mathbf{q}_t^{b_k} dt,
\end{aligned}
\tag{3}
$$

where

$$
\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -\lfloor\boldsymbol{\omega}\rfloor_\times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^T & 0 \end{bmatrix}, \lfloor\boldsymbol{\omega}\rfloor_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}.
\tag{4}
$$

> 这里的公式3的四元数的积分表达式的推导点击这里
>
> 这篇博客先是推导了一下四元数相乘写成左乘和右乘矩阵的形式。
>
> 然后代入导数的定义式，然后假设方向是和当前旋转的方向一致，然后利用等价无穷小进行替换推导
>
> 最终可以得到$\dot{q}_w = \frac{1}{2}\Omega(w)q_w$的形式
>
> 对$\dot{g}(x) = f(x)$,我们有$g(x+\Delta t) = g(x) + \int f(\tau)d\tau$

然后上式的表达式依赖于b_k时刻的速度，位置，和旋转。当初始状态发生改变的时候（例如b_0）时刻更新了，那么这些都要重新计算。计算量太大了，所以整理一下形式，变成了类似Δ的形式。

$$
\begin{aligned}
\mathbf{R}_w^{b_k}\mathbf{p}_{b_{k+1}}^{w} &= \mathbf{R}_w^{b_k}\left(\mathbf{p}_{b_k}^{w} + \mathbf{v}_{b_k}^{w}\Delta t_k - \frac{1}{2}\mathbf{g}^w\Delta t_k^2\right) + \boldsymbol{\alpha}_{b_{k+1}}^{b_k} \\
\mathbf{R}_w^{b_k}\mathbf{v}_{b_{k+1}}^{w} &= \mathbf{R}_w^{b_k}\left(\mathbf{v}_{b_k}^{w} - \mathbf{g}^w\Delta t_k\right) + \boldsymbol{\beta}_{b_{k+1}}^{b_k} \\
\mathbf{q}_w^{b_k} \otimes \mathbf{q}_{b_{k+1}}^{w} &= \boldsymbol{\gamma}_{b_{k+1}}^{b_k},
\end{aligned}
\tag{5}
$$

$$
\begin{aligned}
\boldsymbol{\alpha}_{b_{k+1}}^{b_k} &= \iint_{t\in[t_k,t_{k+1}]} \mathbf{R}_t^{b_k}(\hat{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a)dt^2 \\
\boldsymbol{\beta}_{b_{k+1}}^{b_k} &= \int_{t\in[t_k,t_{k+1}]} \mathbf{R}_t^{b_k}(\hat{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a)dt \\
\boldsymbol{\gamma}_{b_{k+1}}^{b_k} &= \int_{t\in[t_k,t_{k+1}]} \frac{1}{2}\boldsymbol{\Omega}(\hat{\boldsymbol{\omega}}_t - \mathbf{b}_{w_t} - \mathbf{n}_w)\boldsymbol{\gamma}_t^{b_k} dt.
\end{aligned}
\tag{6}
$$

这里的好处是，我们重新计算的时候只需要计算非积分的部分，这部分很快，而积分部分很慢，但是这部分仅仅依赖于加速度$\hat{a}$和角速度$\hat{\omega}$。当bias发生小的变化的时候，我们用一阶泰勒展开去更新就好了，当发生较大变化的时候，我们重新计算。

我们对6的积分用欧拉积分来表示，这里这是便于推导，实际上程序用的是mid-point积分。

$$\begin{aligned}
\hat{\boldsymbol{\alpha}}_{i+1}^{b_k} &= \hat{\boldsymbol{\alpha}}_{i}^{b_k} + \hat{\boldsymbol{\beta}}_{i}^{b_k}\delta t + \frac{1}{2}\mathbf{R}(\hat{\boldsymbol{\gamma}}_{i}^{b_k})(\hat{\mathbf{a}}_i - \mathbf{b}_{a_i})\delta t^2 \\
\hat{\boldsymbol{\beta}}_{i+1}^{b_k} &= \hat{\boldsymbol{\beta}}_{i}^{b_k} + \mathbf{R}(\hat{\boldsymbol{\gamma}}_{i}^{b_k})(\hat{\mathbf{a}}_i - \mathbf{b}_{a_i})\delta t \\
\hat{\boldsymbol{\gamma}}_{i+1}^{b_k} &= \hat{\boldsymbol{\gamma}}_{i}^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}(\hat{\boldsymbol{\omega}}_i - \mathbf{b}_{w_i})\delta t \end{bmatrix}
\end{aligned} \quad . \tag{7}$$

然后因为四元数表示的旋转其实是超过一个自由度了的，所以这里定义了一下$\delta\theta$

$$\boldsymbol{\gamma}_{t}^{b_k} \approx \hat{\boldsymbol{\gamma}}_{t}^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\delta\boldsymbol{\theta}_{t}^{b_k} \end{bmatrix}, \tag{8}$$

其实这里可以将$\delta\theta$理解为一个旋转向量（对，就是李代数）。然后论文就直接给出了积分项的误差的导数表达式了。

$$\begin{aligned}
\begin{bmatrix} \delta\dot{\boldsymbol{\alpha}}_{t}^{b_k} \\ \delta\dot{\boldsymbol{\beta}}_{t}^{b_k} \\ \delta\dot{\boldsymbol{\theta}}_{t}^{b_k} \\ \delta\dot{\mathbf{b}}_{a_t} \\ \delta\dot{\mathbf{b}}_{w_t} \end{bmatrix} &= \begin{bmatrix} 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & -\mathbf{R}_{t}^{b_k}\lfloor\hat{\mathbf{a}}_t - \mathbf{b}_{a_t}\rfloor_\times & -\mathbf{R}_{t}^{b_k} & 0 \\ 0 & 0 & -\lfloor\hat{\boldsymbol{\omega}}_t - \mathbf{b}_{w_t}\rfloor_\times & 0 & -\mathbf{I} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\alpha}_{t}^{b_k} \\ \delta\boldsymbol{\beta}_{t}^{b_k} \\ \delta\boldsymbol{\theta}_{t}^{b_k} \\ \delta\mathbf{b}_{a_t} \\ \delta\mathbf{b}_{w_t} \end{bmatrix} \\
&+ \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\mathbf{R}_{t}^{b_k} & 0 & 0 & 0 \\ 0 & -\mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{n}_a \\ \mathbf{n}_w \\ \mathbf{n}_{b_a} \\ \mathbf{n}_{b_w} \end{bmatrix} = \mathbf{F}_t\delta\mathbf{z}_{t}^{b_k} + \mathbf{G}_t\mathbf{n}_t,
\end{aligned} \tag{9}$$

这里，我有必要推导一下，毕竟花了很多时间，参考了很多资料。

首先我们要定义清楚这些符号的意思，论文里只写了γ的表达式，我这里为了推导方便。全部整理一下

$$\alpha = \hat{\alpha} + \delta\alpha \tag{1}$$

$$\beta = \hat{\beta} + \delta\beta \tag{2}$$

$$\gamma = \hat{\gamma} \otimes [1, \frac{1}{2}\delta\theta] \tag{3}$$

$$\dot{\alpha} = \beta \tag{4}$$

$$\dot{\hat{\alpha}} = \hat{\beta} \tag{5}$$

$$\dot{\beta} = R(\hat{a} - b_a - n_a) \tag{6}$$

$$\dot{\hat{\beta}} = \hat{R}(\hat{a} - \hat{b}_a) \tag{7}$$

$$b_a = \hat{b}_a + \delta b_a = \hat{b} + \int n_{b_a} t dt \tag{8}$$

$$b_w = \hat{b}_w + \delta b_w = \hat{b}_w + \int n_{b_w} t dt \tag{9}$$

利用(1,2,4,5)可以得到

$$\dot{\delta\alpha} = \dot{\alpha} - \dot{\hat{\alpha}} = \beta - \hat{\beta} = \delta\beta$$

对应矩阵的1行。

利用(8)(9)可以的得到

$$\dot{\delta b_a} = n_{b_a}$$
$$\dot{\delta b_w} = n_{b_w}$$

对应矩阵的4、5行。

接下来求一下矩阵的第二行，这里我用的是李代数的方法，个人觉得这个好推导一点。

首先，我们有

$$\begin{aligned} R &= \hat{R}\delta R = RExp(\delta\theta) \\ =>\hat{R} &= R\delta R^{-1} = RExp(-\delta\theta) \overset{\text{一阶泰勒展开}}{=} R(I - \lfloor\delta\theta\rfloor_\times) \end{aligned}$$

$$\begin{aligned} \dot{\delta\beta} =& R(\hat{a} - b_a - n_a) - \hat{R}(\hat{a} - \hat{b}_a) \\ =& R(\hat{a} - b_a - n_a) - R(I - \lfloor\delta\theta\rfloor_\times)(\hat{a} - \hat{b}_a) \\ =& R(\hat{a} - b_a - n_a - \hat{a} + \hat{b}_a + \lfloor\delta\theta\rfloor_\times(\hat{a} - \hat{b}_a)) \\ =& -Rn_a - R(\delta b_a) - R\lfloor\hat{a} - b_a\rfloor_\times\delta\theta + R\lfloor\delta b_a\rfloor_\times\delta\theta \\ \overset{\text{忽略二阶小量}}{=}& -Rn_a - R(\delta b_a) - R\lfloor\hat{a} - b_a\rfloor_\times\delta\theta \end{aligned}$$

这就得到了矩阵的第2行。

接下来求矩阵的第3行

> 这里我尝试着用李代数去推导，但是推不出来，但是用四元数的方法可以。

首先，我们有

$$\mathbf{q}_1 \otimes \mathbf{q}_2 = [\mathbf{q}_1]_L \, \mathbf{q}_2 \qquad \text{and} \qquad \mathbf{q}_1 \otimes \mathbf{q}_2 = [\mathbf{q}_2]_R \, \mathbf{q}_1$$

$$[\mathbf{q}]_L = \begin{bmatrix} q_w & -q_x & -q_y & -q_z \\ q_x & q_w & -q_z & q_y \\ q_y & q_z & q_w & -q_x \\ q_z & -q_y & q_x & q_w \end{bmatrix}, \qquad [\mathbf{q}]_R = \begin{bmatrix} q_w & -q_x & -q_y & -q_z \\ q_x & q_w & q_z & -q_y \\ q_y & -q_z & q_w & q_x \\ q_z & q_y & -q_x & q_w \end{bmatrix}$$

然后，我们有

$$[\mathbf{q}]_L = q_w \mathbf{I} + \begin{bmatrix} 0 & -\mathbf{q}_v^\top \\ \mathbf{q}_v & [\mathbf{q}_v]_\times \end{bmatrix}, \qquad [\mathbf{q}]_R = q_w \mathbf{I} + \begin{bmatrix} 0 & -\mathbf{q}_v^\top \\ \mathbf{q}_v & -[\mathbf{q}_v]_\times \end{bmatrix}$$

$$[\mathbf{a}]_\times \triangleq \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

反对称矩阵的记号

然后，我们有

$$\dot{q} = \frac{1}{2}[0, w]_R q = \frac{1}{2} q \otimes [0, w] \overset{\text{简略记号}}{=} \frac{1}{2} q \otimes w$$

$$\begin{aligned} \dot{\gamma} &= \dot{\hat{\gamma}} \otimes \delta\gamma + \hat{\gamma} \otimes \dot{\delta\gamma} \\ \frac{1}{2}\hat{\gamma} \otimes \delta\gamma \otimes w &= \frac{1}{2}\hat{\gamma} \otimes (\hat{w} - \hat{b}_w) \otimes \delta\gamma + \hat{\gamma} \otimes \dot{\delta\gamma} \\ \dot{\delta\gamma} &= \frac{1}{2}\delta\gamma \otimes w - \frac{1}{2}(\hat{w} - \hat{b}_w) \otimes \delta\gamma \\ [0, \frac{1}{2}\dot{\delta\theta}] &= \frac{1}{2}[w]_R \delta\gamma - \frac{1}{2}[\hat{w} - \hat{b}_w]_L \delta\gamma \end{aligned}$$

将$\dot{\delta\theta}$拿出来，有

$$\begin{aligned} \begin{bmatrix} 0 \\ \dot{\delta\theta} \end{bmatrix} &= \left(\begin{bmatrix} 0 & -w^T \\ w & -\lfloor w \rfloor_\times \end{bmatrix} - \begin{bmatrix} 0 & -(\hat{w} - \hat{b}_w)^T \\ (\hat{w} - \hat{b}_w) & \lfloor \hat{w} - \hat{b}_w \rfloor_\times \end{bmatrix} \right) \begin{bmatrix} 1 \\ \frac{1}{2}\delta\theta \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{2}(\delta b_w + n_w)^T \delta\theta \\ -\delta b_w - n_w + (-\lfloor \hat{w} - b_w \rfloor_\times)\delta\theta + \frac{1}{2}\lfloor \delta b_w + n_w \rfloor_\times \delta\theta \end{bmatrix} \end{aligned}$$

忽略二阶无穷小量，我们最终可以得到

$$\dot{\delta\theta} = -\delta b_w - n_w + (-\lfloor \hat{w} - b_w \rfloor_\times)\delta\theta$$

到此，我们把整个矩阵都证明出来了。

随着预积分的进行，我们可以得到

$$\delta z_{t+\delta t} = \delta z_t + \dot{\delta z_t} \times \delta t = (\delta t F_t + I)\delta z_t + G_t n_t \delta t$$

那这样就很容易得到$t + \delta t$时刻的协方差矩阵了.

$\mathbf{P}_{b_{k+1}}^{b_k}$ can be computed recursively by first-order discrete-time covariance update with the initial covariance $\mathbf{P}_{b_k}^{b_k} = \mathbf{0}$:

$$\mathbf{P}_{t+\delta t}^{b_k} = (\mathbf{I} + \mathbf{F}_t \delta t)\mathbf{P}_t^{b_k}(\mathbf{I} + \mathbf{F}_t \delta t)^T + (\mathbf{G}_t \delta t)\mathbf{Q}(\mathbf{G}_t \delta t)^T, \quad t \in [k, k+1], \tag{10}$$

where $\mathbf{Q}$ is the diagonal covariance matrix of noise $(\boldsymbol{\sigma}_a^2, \boldsymbol{\sigma}_w^2, \boldsymbol{\sigma}_{b_a}^2, \boldsymbol{\sigma}_{b_w}^2)$.

同时也很容易得到$t + \delta t$时刻的雅可比矩阵

$$J_{t+\delta t} = ((\delta t F_t + I)\delta z_t + G_t n_t \delta t)' = (\delta t F_t + I) J_t$$

Meanwhile, the first-order Jacobian matrix $\mathbf{J}_{b_{k+1}}$ of $\delta \mathbf{z}_{b_{k+1}}^{b_k}$ with respect to $\delta \mathbf{z}_{b_k}^{b_k}$ can be also compute recursively with the initial Jacobian $\mathbf{J}_{b_k} = \mathbf{I}$,

$$\mathbf{J}_{t+\delta t} = (\mathbf{I} + \mathbf{F}_t \delta t)\mathbf{J}_t, \quad t \in [k, k+1]. \tag{11}$$

理论上来说，我们可以递归的得到$t \in [k, k+1]$里面的δz_t的协方差矩阵P_t和雅可比矩阵J_t.

> 注意这个雅可比矩阵不是对时间的导数，而是对δz_t的各个分量的导数。

当各个偏差项发生较小的变化的时候，可以用一阶泰勒展开去近似，当发生较大的变化的时候，需要重新预计分。

Using this recursive formulation, we get the covariance matrix $\mathbf{P}_{b_{k+1}}^{b_k}$ and the Jacobian $\mathbf{J}_{b_{k+1}}$. The first order approximation of $\boldsymbol{\alpha}_{b_{k+1}}^{b_k}, \boldsymbol{\beta}_{b_{k+1}}^{b_k}, \boldsymbol{\gamma}_{b_{k+1}}^{b_k}$ with respect to biases can be write as:

$$\begin{aligned}
\boldsymbol{\alpha}_{b_{k+1}}^{b_k} &\approx \hat{\boldsymbol{\alpha}}_{b_{k+1}}^{b_k} + \mathbf{J}_{b_a}^{\alpha} \delta \mathbf{b}_{a_k} + \mathbf{J}_{b_w}^{\alpha} \delta \mathbf{b}_{w_k} \\
\boldsymbol{\beta}_{b_{k+1}}^{b_k} &\approx \hat{\boldsymbol{\beta}}_{b_{k+1}}^{b_k} + \mathbf{J}_{b_a}^{\beta} \delta \mathbf{b}_{a_k} + \mathbf{J}_{b_w}^{\beta} \delta \mathbf{b}_{w_k} \\
\boldsymbol{\gamma}_{b_{k+1}}^{b_k} &\approx \hat{\boldsymbol{\gamma}}_{b_{k+1}}^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2} \mathbf{J}_{b_w}^{\gamma} \delta \mathbf{b}_{w_k} \end{bmatrix}
\end{aligned} \tag{12}$$

where $\mathbf{J}_{b_a}^{\alpha}$ and is the sub-block matrix in $\mathbf{J}_{b_{k+1}}$ whose location is corresponding to $\frac{\delta \boldsymbol{\alpha}_{b_{k+1}}^{b_k}}{\delta \mathbf{b}_{a_k}}$. The same meaning is also used for $\mathbf{J}_{b_w}^{\alpha}, \mathbf{J}_{b_a}^{\beta}, \mathbf{J}_{b_w}^{\beta}, \mathbf{J}_{b_w}^{\gamma}$. When the estimation of bias changes

最终我们可以得到观测模型

$$\begin{bmatrix} \hat{\boldsymbol{\alpha}}_{b_{k+1}}^{b_k} \\ \hat{\boldsymbol{\beta}}_{b_{k+1}}^{b_k} \\ \hat{\boldsymbol{\gamma}}_{b_{k+1}}^{b_k} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_w^{b_k} (\mathbf{p}_{b_{k+1}}^w - \mathbf{p}_{b_k}^w + \frac{1}{2} \mathbf{g}^w \Delta t_k^2 - \mathbf{v}_{b_k}^w \Delta t_k) \\ \mathbf{R}_w^{b_k} (\mathbf{v}_{b_{k+1}}^w + \mathbf{g}^w \Delta t_k - \mathbf{v}_{b_k}^w) \\ \mathbf{q}_{b_k}^{w^{-1}} \otimes \mathbf{q}_{b_{k+1}}^w \\ \mathbf{b}_{a b_{k+1}} - \mathbf{b}_{a b_k} \\ \mathbf{b}_{w b_{k+1}} - \mathbf{b}_{w b_k} \end{bmatrix}. \tag{13}$$

其实这个东西，就是优化的时候的约束边了。

左边可以看成是IMU预积分出来的值，带有噪音。右边是通过某种手段得到的状态变量（例如，VO）。

这一节的思路大概就是为了得到（10）和（12）。10衡量了这个带噪音的积分项的方差，给偏差发生改变的时候，快速更新积分项提供了方便。

ESTIMATOR INITIALIZATION

主要的工作便是要初始化IMU，这里讲讲我认为的初始化的原因和目的

原因和目的

1. 开机的时候，不一定处于静止状态，IMU只能够读取加速度，所以位姿和速度都是通过积分出来的，所以此时我们接下来无法得到速度。
2. IMU存在着固定偏差bais，如果不想办法消除误差，之后也是会越飘越远（想象一下，如果一个小的固定增量，经过长时间的二次积分所带来的影响）

解决方案

采用松耦合的方法获取初始值。通过VO，使用eight-point or five point or estimating the homogeneous之类的方式，我们可以得到一些初值。通过对齐这些数据到IMU在这个期间的预积分的值，我们可以得到IMU的初值。同时，我们在初始化的阶段忽略了IMU的加速度bias，这是因为gravity和bias耦合在了一起，并且gravity会远大于bias，以及在初始化的短时间片段内，很难观测到加速度的bias.

这些初值包括：

- 相机
  - scale
- IMU
  - gravity
  - velocity
  - bias

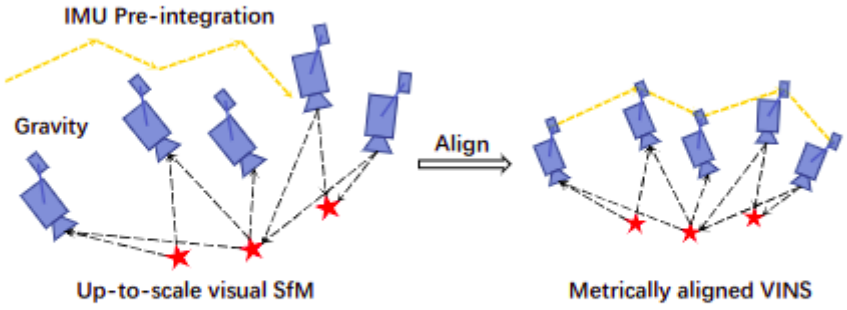


Fig. 4. An illustration of the visual-inertial alignment process for estimator initialization.

具体的初始化过程其实是分成了两步.

1. 获取VO信息.

- 维持着一个slide windows。当如果当前帧和窗口里面的任意一帧满足一定的参数需求的时候(多于30个匹配点对同时略去旋转误差之后还有20个像素以上的的视差)，则用五点法评估两帧之间的相对关系。
- 在获得相对的运动关系之后，使用一个任意的scale（之后会求解出来）,配合三角测量，得到当前帧的特征点的3D坐标，然后连续使用Pnp求解出整个窗口里面的相对位姿，然后用BA去最小化重投影误差.到这里我们已经得到了VO的一段轨迹了。

- 由于我们没有对世界的任何先验，所以我们将窗口的第一帧设置为参考帧。

we do not yet have any knowledge about the world frame, we set the first camera frame $(\cdot)^{c_0}$ as the reference frame for SfM. All frame poses $(\bar{\mathbf{p}}_{c_k}^{c_0}, \mathbf{q}_{c_k}^{c_0})$ and feature positions are represented with respect to $(\cdot)^{c_0}$. Suppose we have a rough measure extrinsic parameters $(\mathbf{p}_c^b, \mathbf{q}_c^b)$ between the camera and the IMU, we can translate poses from camera frame to body (IMU) frame,

$$\begin{aligned} \mathbf{q}_{b_k}^{c_0} &= \mathbf{q}_{c_k}^{c_0} \otimes (\mathbf{q}_c^b)^{-1} \\ s\bar{\mathbf{p}}_{b_k}^{c_0} &= s\bar{\mathbf{p}}_{c_k}^{c_0} - \mathbf{R}_{b_k}^{c_0}\mathbf{p}_c^b, \end{aligned} \tag{14}$$

where s the scaling parameter that aligns the visual structure to the metric scale. Solving this scaling parameter is the key to achieve successful initialization.

这里，相机和IMU的相对关系是**粗略测量的。**（14）是表示的是b_k时刻的IMU在c_0这个frame下的位姿。可以看成$p_{b_k}^{c_0} = s\bar{p}_{b_k}^{c_0}$

2.对齐IMU和VO的数据

1) 获取b_w

其实就是我们通过VO知道了连续两帧之间的旋转（注意，旋转是没有尺度问题的),同时我们也知道了IMU关于姿态的预计积分变化（注意，IMU读取的是角速度，不是角加速度，所以积分结果直接就是姿态的变化值）。然后理论上这两者是要相等的，这里我们假设VO是准的，那么预计分与VO的结果之间的差异自然就是由b_w所导致的了。

1) Gyroscope Bias Calibration: Consider two consecutive frames b_k and b_{k+1} in the window, we get the rotation $\mathbf{q}_{b_k}^{c_0}$ and $\mathbf{q}_{b_{k+1}}^{c_0}$ from the visual SfM, as well as the relative constraint $\hat{\gamma}_{b_{k+1}}^{b_k}$ from IMU pre-integration. We linearize the IMU pre-integration term with respect to gyroscope bias and minimize the following cost function:

$$\begin{aligned} \min_{\delta b_w} \sum_{k\in\mathcal{B}} \left\| {\mathbf{q}_{b_{k+1}}^{c_0}}^{-1} \otimes \mathbf{q}_{b_k}^{c_0} \otimes \boldsymbol{\gamma}_{b_{k+1}}^{b_k} \right\|^2 \\ \boldsymbol{\gamma}_{b_{k+1}}^{b_k} \approx \hat{\boldsymbol{\gamma}}_{b_{k+1}}^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\mathbf{J}_{b_w}^{\gamma}\delta\mathbf{b}_w \end{bmatrix}, \end{aligned} \tag{15}$$

where $\mathcal{B}$ indexes all frames in the window. We have the first order approximation of $\hat{\gamma}_{b_{k+1}}^{b_k}$ with respect to the gyroscope bias using the bias Jacobian derived in Sect. IV-B. In such way, we get an initial calibration of the gyroscope bias $\mathbf{b}_w$. Then we re-propagate all IMU pre-integration terms $\hat{\boldsymbol{\alpha}}_{b_{k+1}}^{b_k}, \hat{\boldsymbol{\beta}}_{b_{k+1}}^{b_k}$, and $\hat{\gamma}_{b_{k+1}}^{b_k}$ using the new gyroscope bias.

2) 获取速度、重力向量和尺度

other essential states for navigation, namely velocity, gravity vector, and metric scale:

$$\mathcal{X}_I = \left[\mathbf{v}_{b_0}^{b_0}, \mathbf{v}_{b_1}^{b_1}, \cdots \mathbf{v}_{b_n}^{b_n}, \mathbf{g}^{c_0}, s\right], \tag{16}$$

where $\mathbf{v}_{b_k}^{b_k}$ is velocity in the body frame while taking the k^{th} image, $\mathbf{g}^{c_0}$ is the gravity vector in the c_0 frame, and s scales the monocular SfM to metric units.

注意这里的$v_{b_k}^{b_k}$上标指的是在bodyframe里面取第k张图片的时候的瞬时速度。

对任意两个连续的两帧，我们都可以写出这条表达式，根据式子（5）

Consider two consecutive frames b_k and b_{k+1} in the window, then (5) can be written as:

$$\begin{aligned} \boldsymbol{\alpha}_{b_{k+1}}^{b_k} &= \mathbf{R}_{c_0}^{b_k}\left(s(\bar{\mathbf{p}}_{b_{k+1}}^{c_0} - \bar{\mathbf{p}}_{b_k}^{c_0}) + \frac{1}{2}\mathbf{g}^{c_0}\Delta t_k^2 - \mathbf{R}_{b_k}^{c_0}\mathbf{v}_{b_k}^{b_k}\Delta t_k\right) \\ \boldsymbol{\beta}_{b_{k+1}}^{b_k} &= \mathbf{R}_{c_0}^{b_k}\left(\mathbf{R}_{b_{k+1}}^{c_0}\mathbf{v}_{b_{k+1}}^{b_{k+1}} + \mathbf{g}^{c_0}\Delta t_k - \mathbf{R}_{b_k}^{c_0}\mathbf{v}_{b_k}^{b_k}\right). \end{aligned} \tag{17}$$

然后结合14 17可以得到

We can combine (14) and (17) into the following linear measurement model:

$$\hat{\mathbf{z}}^{b_k}_{b_{k+1}} = \begin{bmatrix} \hat{\boldsymbol{\alpha}}^{b_k}_{b_{k+1}} - \mathbf{p}^b_c + \mathbf{R}^{b_k}_{c_0}\mathbf{R}^{c_0}_{b_{k+1}}\mathbf{p}^b_c \\ \hat{\boldsymbol{\beta}}^{b_k}_{b_{k+1}} \end{bmatrix} = \mathbf{H}^{b_k}_{b_{k+1}}\mathcal{X}_I + \mathbf{n}^{b_k}_{b_{k+1}} \tag{18}$$

where,

$$\mathbf{H}^{b_k}_{b_{k+1}} = \begin{bmatrix} -\mathbf{I}\Delta t_k & \mathbf{0} & \frac{1}{2}\mathbf{R}^{b_k}_{c_0}\Delta t_k^2 & \mathbf{R}^{b_k}_{c_0}(\bar{\mathbf{p}}^{c_0}_{c_{k+1}} - \bar{\mathbf{p}}^{c_0}_{c_k}) \\ -\mathbf{I} & \mathbf{R}^{b_k}_{c_0}\mathbf{R}^{c_0}_{b_{k+1}} & \mathbf{R}^{b_k}_{c_0}\Delta t_k & \mathbf{0} \end{bmatrix} \tag{19}$$

注意一下19的最后一列的下角标不再是b而是变成了c，这也是为什么18左边的第一行多了两项看起来有点奇怪的东西，不仔细看还真看不出来。

在18这条观测方程里面，左边的部分是IMU预积分和外参组成，右边的H矩阵由VO提供，$\mathcal{X}_I$是待求变量，n是噪声项，我们假设为0就是了。然后就可以得到了一个最小二乘的表达式，只要解开就能得到我们的代求变量。

It can be seen that $\mathbf{R}^{c_0}_{b_k}, \mathbf{R}^{c_0}_{b_{k+1}}, \bar{\mathbf{p}}^{c_0}_{c_k}, \bar{\mathbf{p}}^{c_0}_{c_{k+1}}$ are obtained from the up-to-scale monocular visual SfM. Δt_k is the time interval between two consecutive frames. By solving this linear least square problem:

$$\min_{\mathcal{X}_I} \sum_{k\in\mathcal{B}} \left\| \hat{\mathbf{z}}^{b_k}_{b_{k+1}} - \mathbf{H}^{b_k}_{b_{k+1}}\mathcal{X}_I \right\|^2, \tag{20}$$

we can get body frame velocities for every frame in the window, the gravity vector in the visual reference frame $(\cdot)^{c_0}$, as well as the scale parameter.

5)精炼一下重力项

> 在看vins这篇论文的时候，真的觉得，作者是为了把自己能想到的东西，以及知道的信息都尽量的加到论文里，同时一直也在考虑工程的运行效率之类的问题。

这部分主要的想法就是，既然我们已知了重力的大小，那么就把重力项分解为三个正交的基底的线性组成，同时，要保证其中一个基底的分量为$g = 9.81$。我们不断的更新g的方向，g每次一更新就同时更新一下剩下的基底的方向。知道g的方向收敛。

找基底其实很简单，运用叉乘运算就好了，注意，一个三维的正交基底，我们确定一个轴后，还是有无穷都的可能性的，它这里比较随意，无所谓，只要满足就好了。

Algorithm 1: Finding $\mathbf{b}_1$ and $\mathbf{b}_2$

if $\bar{\hat{\mathbf{g}}} \neq [1,0,0]$ **then**
  $\mathbf{b}_1 \leftarrow normalize(\bar{\hat{\mathbf{g}}} \times [1,0,0])$;
else
  $\mathbf{b}_1 \leftarrow normalize(\bar{\hat{\mathbf{g}}} \times [0,0,1])$;
end
$\mathbf{b}_2 \leftarrow \bar{\hat{\mathbf{g}}} \times \mathbf{b}_1$;

4)完成初始化

既然我们最终会得到重力的方向，那么我们显然能够得到一个z轴与重力方向平行的姿态，我们将这个姿态叫做$q_{c_0}^{w}$。所以我们能够将所有的数据都转化到世界坐标下了（所谓的世界坐标就是以初始化成功的那一帧的位置为原点，以垂直向上为z轴的右手坐标系）。

VI TIGHTLY-COUPLED MONOCULAR VIO

开始紧耦合了

A 公式化

我们待优化的变量为

The full state vector in the sliding window is defined as:

$$\begin{aligned}\mathcal{X} &= \left[\mathbf{x}_0, \mathbf{x}_1, \cdots \mathbf{x}_n, \mathbf{x}_c^b, \lambda_0, \lambda_1, \cdots \lambda_m\right] \\ \mathbf{x}_k &= \left[\mathbf{p}_{b_k}^w, \mathbf{v}_{b_k}^w, \mathbf{q}_{b_k}^w, \mathbf{b}_a, \mathbf{b}_g\right], k \in [0,n] \\ \mathbf{x}_c^b &= \left[\mathbf{p}_c^b, \mathbf{q}_c^b\right],\end{aligned} \tag{21}$$

where $\mathbf{x}_k$ is the IMU state at the time that the k^{th} image is captured. It contains position, velocity, and orientation of the IMU in the world frame, and acceleration bias and gyroscope bias in the IMU body frame. n is the total number of keyframes, and m is the total number of features in the sliding window. λ_l is the inverse depth of the l^{th} feature from its first observation.

可以看到他认为的IMU偏差是固定的，理论上随着时间发生的那点变化应该是归结到了噪音里面了，而噪音部分就用协方差去处理就好了。

$$\min_{\mathcal{X}} \left\{ \|\mathbf{r}_p - \mathbf{H}_p\mathcal{X}\|^2 + \sum_{k\in\mathcal{B}} \left\| \mathbf{r}_{\mathcal{B}}(\hat{\mathbf{z}}_{b_{k+1}}^{b_k}, \mathcal{X}) \right\|^2_{\mathbf{P}_{b_{k+1}}^{b_k}} + \sum_{(l,j)\in\mathcal{C}} \rho(\left\| \mathbf{r}_{\mathcal{C}}(\hat{\mathbf{z}}_l^{c_j}, \mathcal{X}) \right\|^2_{\mathbf{P}_l^{c_j}}) \right\}, \tag{22}$$

where the Huber norm [37] is defined as:

$$\rho(s) = \begin{cases} 1 & s \geq 1, \\ 2\sqrt{s} - 1 & s < 1. \end{cases} \tag{23}$$

所谓的紧耦合，其实就是22这条表达式。第一项是所谓的先验项，先不管。第二项是滑动窗里面的IMU误差，第三项是相机相关的残差了。

B IMU测量误差

$$\begin{aligned} \mathbf{r}_{\mathcal{B}}(\hat{\mathbf{z}}_{b_{k+1}}^{b_k}, \mathcal{X}) &= \begin{bmatrix} \delta\boldsymbol{\alpha}_{b_{k+1}}^{b_k} \\ \delta\boldsymbol{\beta}_{b_{k+1}}^{b_k} \\ \delta\boldsymbol{\theta}_{b_{k+1}}^{b_k} \\ \delta\mathbf{b}_a \\ \delta\mathbf{b}_g \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{R}_w^{b_k}(\mathbf{p}_{b_{k+1}}^w - \mathbf{p}_{b_k}^w + \frac{1}{2}\mathbf{g}^w\Delta t_k^2 - \mathbf{v}_{b_k}^w\Delta t_k) - \hat{\boldsymbol{\alpha}}_{b_{k+1}}^{b_k} \\ \mathbf{R}_w^{b_k}(\mathbf{v}_{b_{k+1}}^w + \mathbf{g}^w\Delta t_k - \mathbf{v}_{b_k}^w) - \hat{\boldsymbol{\beta}}_{b_{k+1}}^{b_k} \\ 2\left[\mathbf{q}_{b_k}^{w\,-1} \otimes \mathbf{q}_{b_{k+1}}^w \otimes (\hat{\boldsymbol{\gamma}}_{b_{k+1}}^{b_k})^{-1}\right]_{xyz} \\ \mathbf{b}_{ab_{k+1}} - \mathbf{b}_{ab_k} \\ \mathbf{b}_{wb_{k+1}} - \mathbf{b}_{wb_k} \end{bmatrix}, \end{aligned} \tag{24}$$

where $[\cdot]_{xyz}$ extracts the vector part of a quaternion $\mathbf{q}$ for error state representation. $\delta\boldsymbol{\theta}_{b_{k+1}}^{b_k}$ is the three dimensional error-state representation of quaternion. $[\hat{\boldsymbol{\alpha}}_{b_{k+1}}^{b_k}, \hat{\boldsymbol{\beta}}_{b_{k+1}}^{b_k}, \hat{\boldsymbol{\gamma}}_{b_{k+1}}^{b_k}]^{\mathrm{T}}$ are pre-integrated IMU measurement terms using only noisy accelerometer and gyroscope measurements within the time interval between two consecutive image frames. Accelerometer and gyroscope biases are also included in the residual terms for online correction.

要理解的话就是，用待优化变量和IMU预积分的值建立一条待优化的边。

C 视觉残差

$$
\begin{aligned}
\mathbf{r}_{\mathcal{C}}(\hat{\mathbf{z}}_l^{c_j}, \mathcal{X}) &= \begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2\end{bmatrix}^T \cdot (\hat{\bar{\mathcal{P}}}_l^{c_j} - \frac{\mathcal{P}_l^{c_j}}{\|\mathcal{P}_l^{c_j}\|}) \\
\hat{\bar{\mathcal{P}}}_l^{c_j} &= \pi_c{}^{-1}(\begin{bmatrix}\hat{u}_l^{c_j} \\ \hat{v}_l^{c_j}\end{bmatrix}) \\
\mathcal{P}_l^{c_j} &= \mathbf{R}_b^c(\mathbf{R}_w^{b_j}(\mathbf{R}_{b_i}^w(\mathbf{R}_c^b \frac{1}{\lambda_l}\pi_c{}^{-1}(\begin{bmatrix}u_l^{c_i} \\ v_l^{c_i}\end{bmatrix}) \\
&\qquad + \mathbf{p}_c^b) + \mathbf{p}_{b_i}^w - \mathbf{p}_{b_j}^w) - \mathbf{p}_c^b),
\end{aligned}
\tag{25}
$$

这里用的不是传统的针孔模型，而是单位球模型，这是为了适应更多的场景。

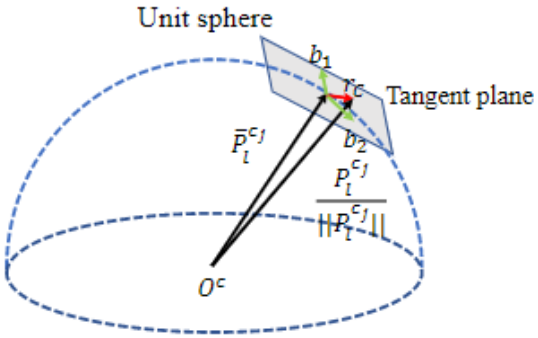


Fig. 6. An illustration of the visual residual on a unit sphere. $\hat{\bar{\mathcal{P}}}_l^{c_j}$ is the unit vector for the observation of the l^{th} feature in the j^{th} frame. $\mathcal{P}_l^{c_j}$ is predicted feature measurement on the unit sphere by transforming its first observation in the i^{th} frame to the j^{th} frame. The residual is defined on the tangent plane of $\hat{\bar{\mathcal{P}}}_l^{c_j}$.

这里的残差在代码实现上有两种形式

- 一种是比较常见的冲投影模型。
- 另外一种是用球面模型，寻找到一组和$\mathcal{P}_l^{c_j}$正交的基底b_1, b_2，然后将r_c投影到这组基底上。

D 边缘化

这一部分主要就是讲一下边缘化的策略。略。看代码，有想法再回来填坑。

E V-I优化策略

主要就是如果每次一个新的帧进来，就优化窗口里面的东西，会太慢了，所以只优化窗口里面最新的一部分，剩下的当作常量，这样子运行效率就能实时了。

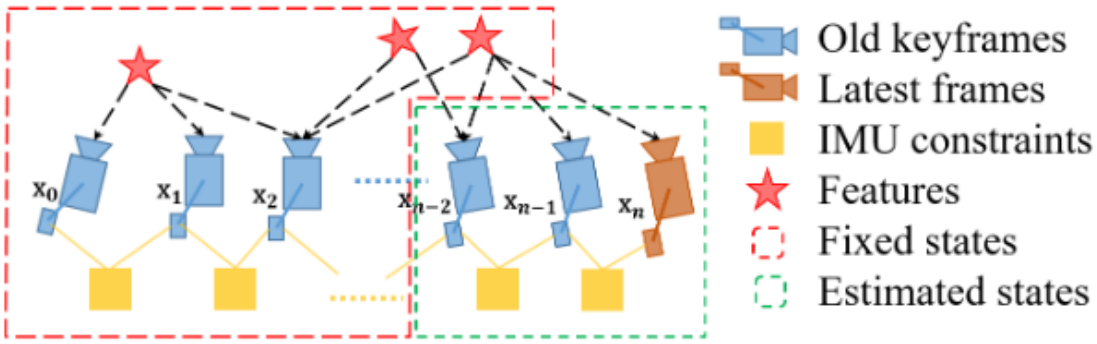


Fig. 8. An illustration of motion-only bundle adjustment for camera-rate outputs.

F IMU传播

虽然帧率受限于相机的表现，但是我们可以利用最新的IMU数据，得到一个IMU帧率一样高的姿态输出频率。

G 失败

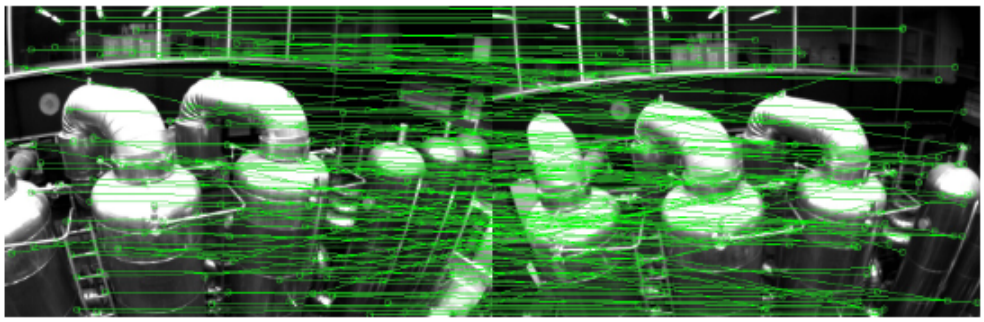
虽然用了IMU和相机，但是理论上还是会存在失败的时候，反正就是他有检测失败的策略，失败了就重新开始。

> 这应该是为了性能上和代码实现上的考虑。工程的方案还是参考一下orb-slam3.

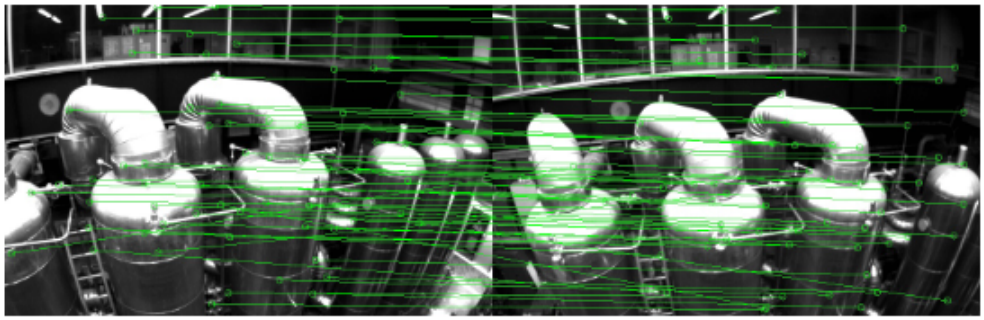
RELOCALIZATION

核心问题就是，虽然你是两个传感器紧耦合，但是还是免不了会飘。但是这个vins由于把重力也一直在评估，所以最后漂的只有四个自由度，xyz+yaw。基于这个东西。他就可以舍弃特征点的旋转不变性，也就是说直接使用BRIEF作为描述子来表示一个特征点。然后基于此用DBOW实现了回环检测和重定位的功能。

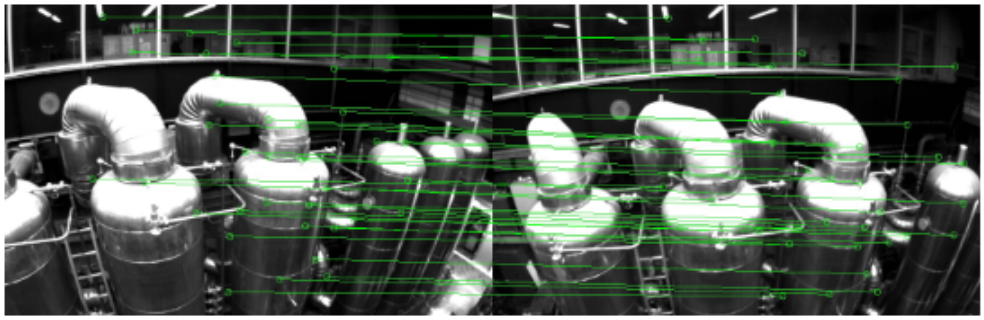
为了匹配更加准确，毕竟回环边的影响很大，如果错了就很不好了，所以用了如下的评估方式。先用fundamental matrix+RANSAC过滤一次，再用pnp+RANSAC过滤一次。剩下的才是要的。



(a) BRIEF descriptor matching results



(b) First step: 2D-2D outlier rejection results



(c) Second step: 3D-2D outlier rejection results.

Fig. 10. Descriptor matching and outlier removal for feature retrieval during loop closure.

加入回环触发的时候，其实整个代价函数也就是多了一项而已。

$$\begin{aligned}\hat{\mathbf{p}}_{ij}^{i} &= \hat{\mathbf{R}}_{i}^{w\,-1}(\hat{\mathbf{p}}_{j}^{w} - \hat{\mathbf{p}}_{i}^{w}) \\ \hat{\psi}_{ij} &= \hat{\psi}_{j} - \hat{\psi}_{i}.\end{aligned} \tag{27}$$

这一项和连续帧之间的残差计算方式没有什么不同。

GLOBAL POSE GRAPH OPTIMIZATION

这个感觉也没什么好说，唯一不太一样的是，这里使用了4个自由度组成了残差而已。而且只和关键帧有关系。

$$\begin{aligned}\hat{\mathbf{p}}_{ij}^{i} &= \hat{\mathbf{R}}_{i}^{w\,-1}(\hat{\mathbf{p}}_{j}^{w} - \hat{\mathbf{p}}_{i}^{w}) \\ \hat{\psi}_{ij} &= \hat{\psi}_{j} - \hat{\psi}_{i}.\end{aligned} \tag{27}$$

$$\mathbf{r}_{i,j}(\mathbf{p}_{i}^{w}, \psi_{i}, \mathbf{p}_{j}^{w}, \psi_{j}) = \begin{bmatrix} \mathbf{R}(\hat{\phi}_{i}, \hat{\theta}_{i}, \psi_{i})^{-1}(\mathbf{p}_{j}^{w} - \mathbf{p}_{i}^{w}) - \hat{\mathbf{p}}_{ij}^{i} \\ \psi_{j} - \psi_{i} - \hat{\psi}_{ij} \end{bmatrix}, \tag{28}$$

$$\min_{\mathbf{p},\psi} \left\{ \sum_{(i,j)\in\mathcal{S}} \|\mathbf{r}_{i,j}\|^{2} + \sum_{(i,j)\in\mathcal{L}} \rho(\|\mathbf{r}_{i,j}\|^{2}) \right\}, \tag{29}$$

顺便它为了让这个图不要太大，有一个策略去管理keyframe，说白了就是去掉比较近的，和比较相似的。

论文算是看完了，剩下的是代码