S-MSCKF 论文公式推导与代码解析

高洪臣

2019 年 9 月 1 日

目录

1 概述

1.1 MSCKF

MSCKF 全称 Multi-State Constraint Kalman Filter（多状态约束下的 Kalman 滤波器），是一种基于滤波的 VIO 算法，2007 年由明尼苏达州大学 Mourikis 在 [1] 中首次提出。MSCKF 在 EKF 框架下融合 IMU 和视觉信息，相较于单纯的 VO 算法，MSCKF 能够适应更剧烈的运动、一定时间的纹理缺失等，具有更高的鲁棒性；相较于基于优化的 VIO 算法（VINS，OKVIS），MSCKF 精度相当，速度更快，适合在计算资源有限的嵌入式平台运行。在机器人、无人机、AR/VR 领域，MSCKF 都有较为广泛的运用，如 Google Project Tango 就用了 MSCKF 进行位姿估计。

1.2 MSCKF vs EKF-SLAM

在传统的 EKF-SLAM 框架中，特征点的信息会加入到特征向量和协方差矩阵里，这种方法的缺点是特征点的信息会给一个初始深度和初始协方差，如果不正确的话，极容易导致后面不收敛，出现 inconsistent 的情况。MSCKF 维护一个 pose 的 FIFO，按照时间顺序排列，可以称为滑动窗口，一个特征点在滑动窗口的几个位姿都被观察到的话，就会在这几个位姿间建立约束，从而进行 KF 的更新。 [2]

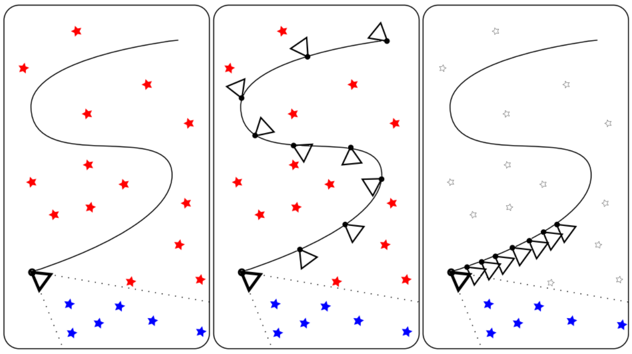
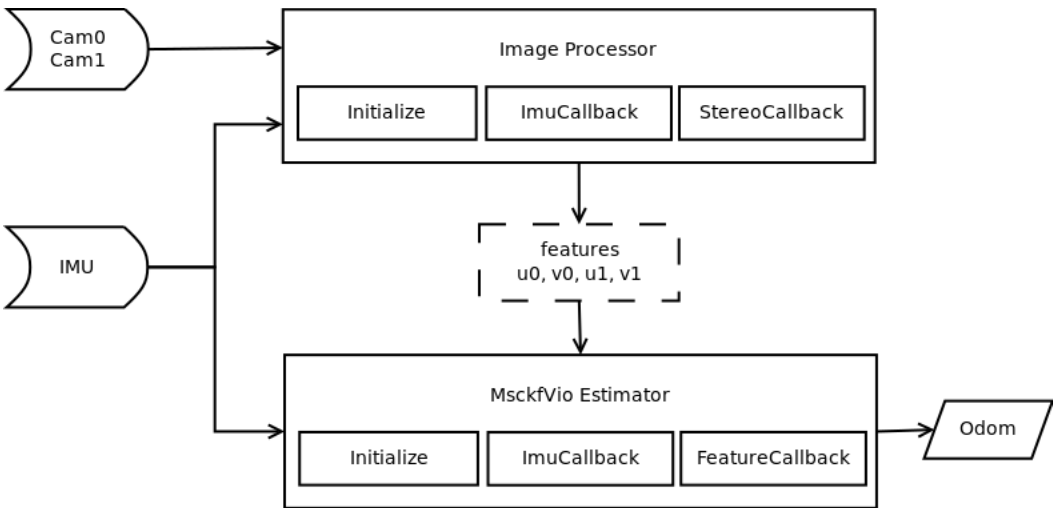


图 1 EKF-SLAM, keyframe-based SLAM, MSCKF

1.3 S-MSCKF

S-MSCKF [3] 是宾夕法尼亚大学 Vijay Kumar 实验室开源的双目版本 MSCKF 算法。



2 Image Processor

2.1 代码流程

2.1.1 Initialize

- load parameters
- create `FastFeatureDetector`

2.1.2 ImuCallback

第一帧图像后，不断添加 IMU message 到 `imu_msg_buffer`。

2.1.3 StereoCallback

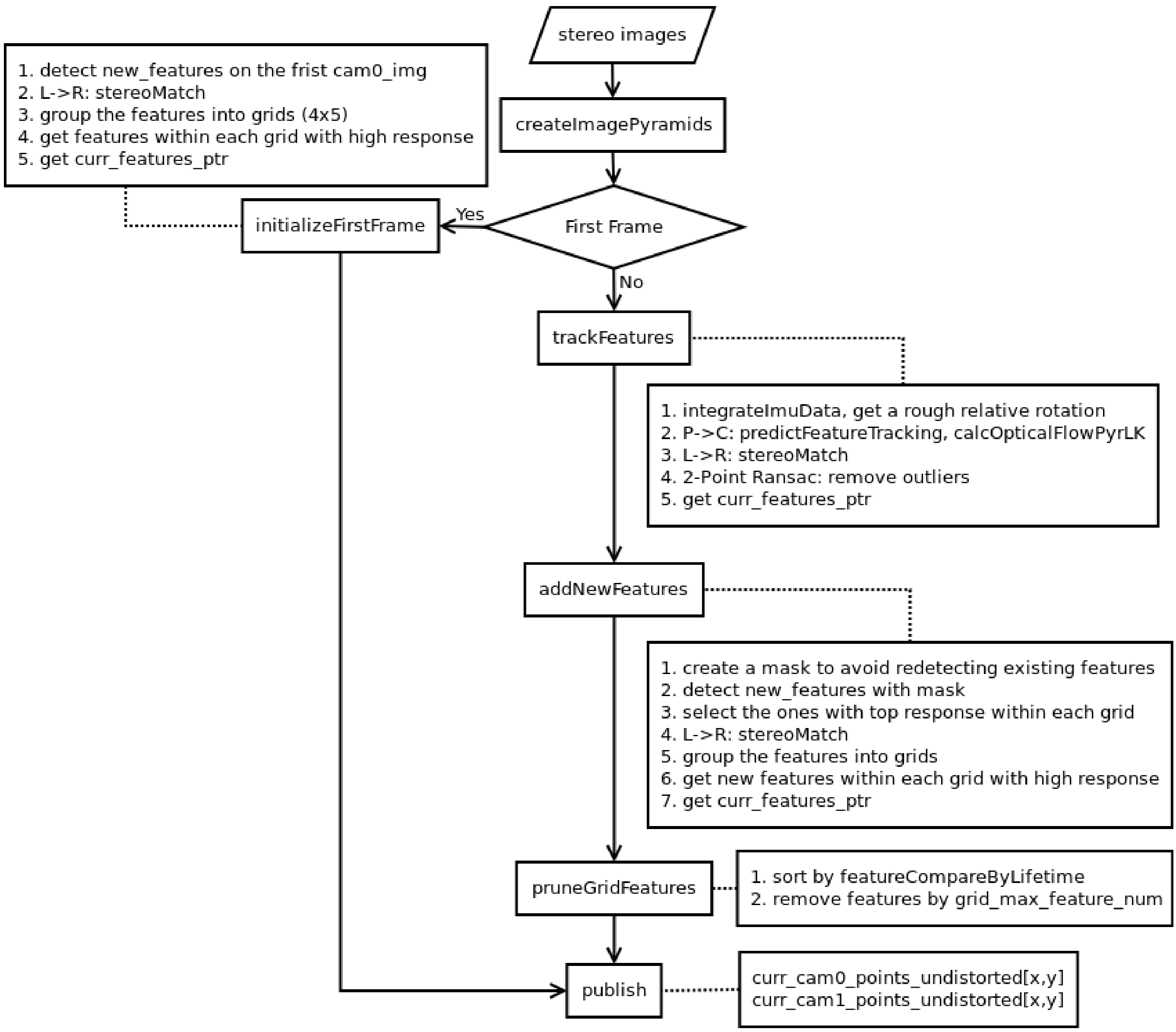


图 2 StereoCallback 流程图

3 MsckfVio Estimator/Filter

3.1 Kalman Filter Overview

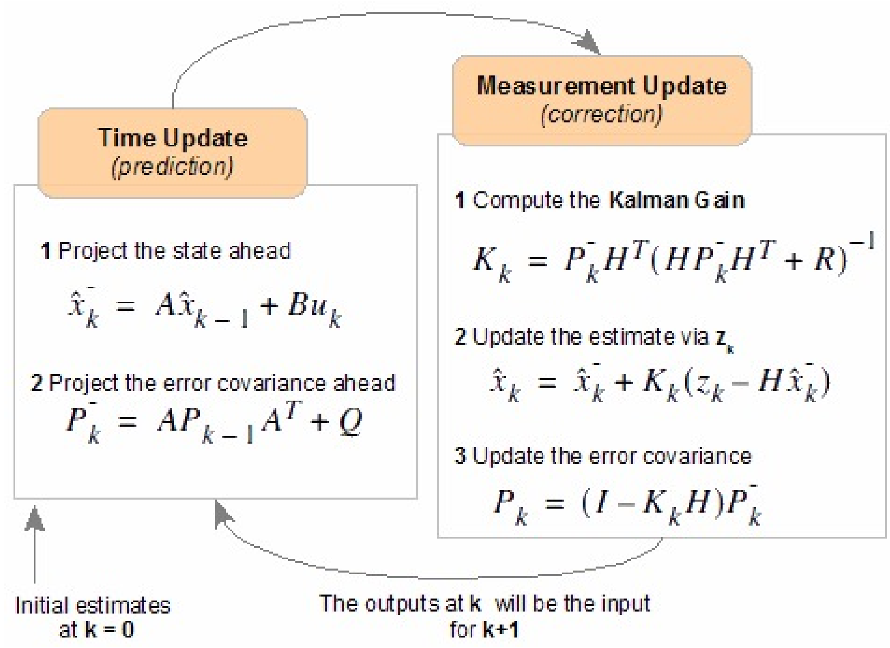


图 3 Kalman Filter 流程图

3.2 代码流程

3.2.1 Initialize

- Load Parameters
- Initialize state server: `state_server.continuous_noise_cov`
- Initialize the chi squared test table with confidence level 0.95: `chi_squared_test_table`

3.2.2 ImuCallback

- 添加 IMU message 到 `imu_msg_buffer`
- `initializeGravityAndBias`，要求前 200 帧 IMU 静止不动
  - 将前 200 帧加速度和角速度求平均
  - 平均加速度的模值 g 作为重力加速度：`IMUState::gravity`
  - 平均角速度作为陀螺仪的 bias：`state_server.imu_state.gyro_bias`
  - 计算重力向量 (0,0,-g) 和平均加速度之间的夹角 (旋转四元数), 标定初始时刻 IMU 系与 world 系之间的夹角：`state_server.imu_state.orientation`

3.2.3 FeatureCallback

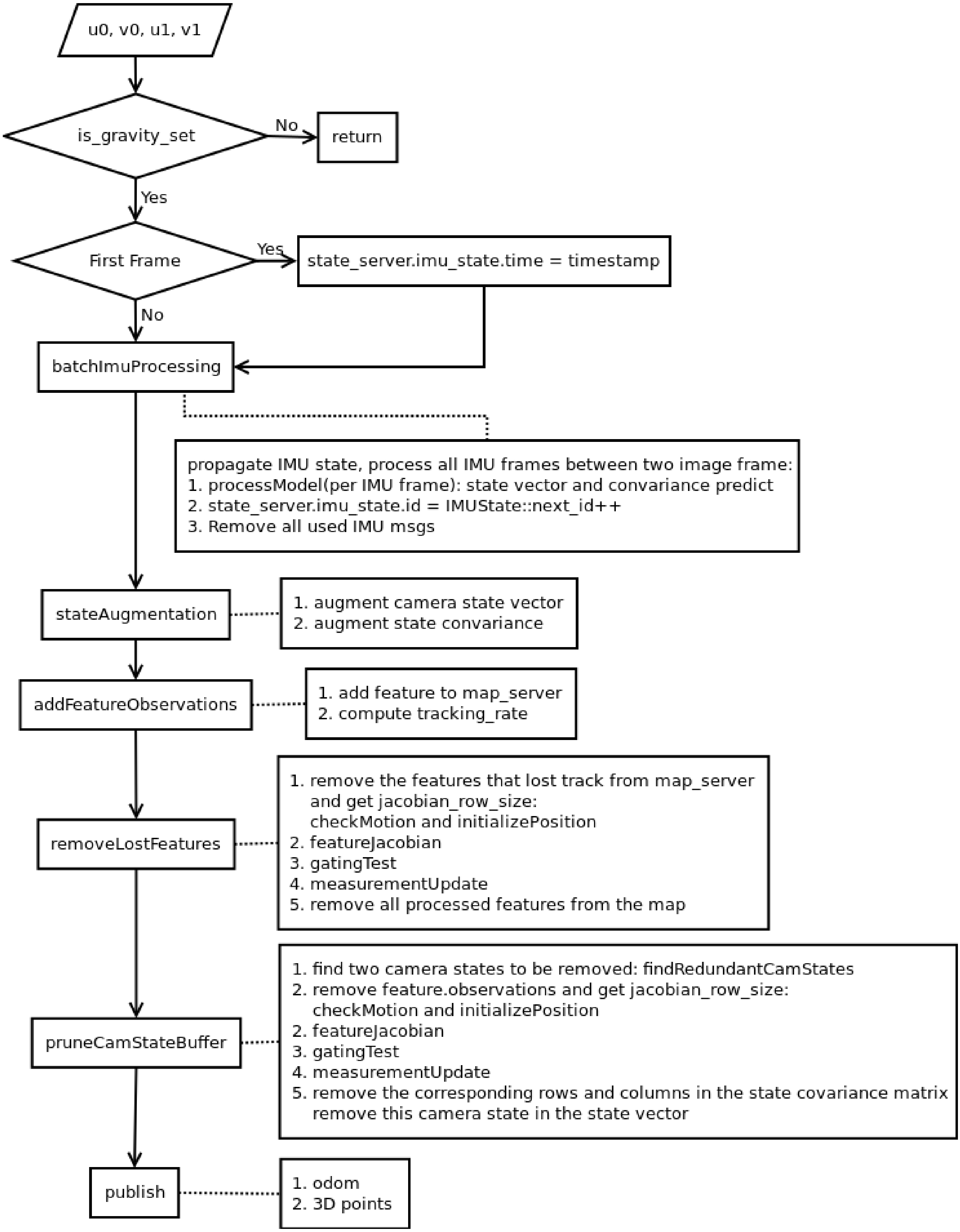


图 4 FeatureCallback 流程图

3.3 EKF 状态向量

IMU 状态向量（true-state）

$$\mathbf{x}_I = \begin{pmatrix} {}^I_G\mathbf{q}^\top & \mathbf{b}_g^\top & {}^G\mathbf{v}_I^\top & \mathbf{b}_a^\top & {}^G\mathbf{p}_I^\top & {}^I_C\mathbf{q}^\top & {}^I\mathbf{p}_C^\top \end{pmatrix}^\top \in \mathbb{R}^{22\times 1}$$

IMU 误差状态向量

$$\tilde{\mathbf{x}}_I = \begin{pmatrix} {}^I_G\tilde{\boldsymbol{\theta}}^\top & \tilde{\mathbf{b}}_g^\top & {}^G\tilde{\mathbf{v}}_I^\top & \tilde{\mathbf{b}}_a^\top & {}^G\tilde{\mathbf{p}}_I^\top & {}^I_C\tilde{\boldsymbol{\theta}}^\top & {}^I\tilde{\mathbf{p}}_C^\top \end{pmatrix}^\top \in \mathbb{R}^{21\times 1}$$

IMU 噪声向量

$$\mathbf{n}_I^\top = \begin{pmatrix} \mathbf{n}_g^\top \ \mathbf{n}_{wg}^\top \ \mathbf{n}_a^\top \ \mathbf{n}_{wa}^\top \end{pmatrix}^\top \in \mathbb{R}^{12\times 1}$$

其对应的噪声协方差矩阵

$$\mathbf{Q}_I = \mathbb{E}\left[\mathbf{n}_I \mathbf{n}_I^\top\right] \in \mathbb{R}^{12\times 12}$$

Camera 误差状态向量

$$\tilde{\mathbf{x}}_{C_i} = \begin{pmatrix} {}^{C_i}_G\tilde{\boldsymbol{\theta}}^\top & {}^G\tilde{\mathbf{p}}_{C_i}^\top \end{pmatrix}^\top \in \mathbb{R}^{6\times 1}$$

系统 (Imu+Camera) 误差状态向量

$$\tilde{\mathbf{x}} = \begin{pmatrix} \tilde{\mathbf{x}}_I^\top & \tilde{\mathbf{x}}_{C_1}^\top & \cdots & \tilde{\mathbf{x}}_{C_N}^\top \end{pmatrix}^\top \in \mathbb{R}^{(21+6N)\times 1}$$

3.4 Propagation/Prediction

3.4.1 IMU 误差状态方程

IMU 运动微分方程（nominal-state 状态方程）

$$\hat{\mathbf{x}} = \begin{cases} {}^I_G\dot{\hat{\mathbf{q}}} = \frac{1}{2}\Omega(\hat{\boldsymbol{\omega}}){}^I_G\hat{\mathbf{q}} \\ \dot{\hat{\mathbf{b}}}_g = \mathbf{0}_{3\times 1} \\ {}^G\dot{\hat{\mathbf{v}}} = C\left({}^I_G\hat{\mathbf{q}}\right)^\top \hat{\mathbf{a}} + {}^G\mathbf{g} \\ \dot{\hat{b}}_a = \mathbf{0}_{3\times 1}, \\ {}^G\dot{\hat{\mathbf{p}}}_I = {}^G\hat{\mathbf{v}} \\ {}^I_C\dot{\hat{\mathbf{q}}} = \mathbf{0}_{3\times 1} \\ {}^I\dot{\hat{\mathbf{p}}}_C = \mathbf{0}_{3\times 1} \end{cases} \tag{1}$$

其中，

$$\hat{\boldsymbol{\omega}} = \boldsymbol{\omega}_m - \hat{\mathbf{b}}_g, \quad \hat{\mathbf{a}} = \mathbf{a}_m - \hat{\mathbf{b}}_a$$

并且

$$\Omega\left(\hat{\boldsymbol{\omega}}\right) \triangleq \begin{bmatrix} \hat{\boldsymbol{\omega}} \\ 0 \end{bmatrix}_R = \begin{pmatrix} -[\hat{\boldsymbol{\omega}}_\times] & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^\top & 0 \end{pmatrix}$$

根据 [4] ESKF 中 ***5.3.3 The error-state kinematics*** 小节公式，IMU error-state 状态方程

$$\delta \dot{\mathbf{x}}_I = \begin{cases} \delta \dot{\mathbf{p}} = \delta \mathbf{v} \\ \delta \dot{\mathbf{v}} = -\mathbf{R}\left[\mathbf{a}_m - \mathbf{a}_b\right]_\times \delta\boldsymbol{\theta} - \mathbf{R}\delta\mathbf{a}_b + \delta\mathbf{g} - \mathbf{R}\mathbf{a}_n \\ \delta\dot{\boldsymbol{\theta}} = -\left[\boldsymbol{\omega}_m - \boldsymbol{\omega}_b\right]_\times \delta\boldsymbol{\theta} - \delta\boldsymbol{\omega}_b - \boldsymbol{\omega}_n \\ \delta\dot{\mathbf{a}}_b = \mathbf{a}_w \\ \delta\dot{\boldsymbol{\omega}}_b = \boldsymbol{\omega}_w \end{cases} \tag{2}$$

对式 (2)线性化，得到 **IMU 连续形式误差状态方程**

$$\dot{\tilde{\mathbf{x}}}_I = \mathbf{F}\tilde{\mathbf{x}}_I + \mathbf{G}\mathbf{n}_I \tag{3}$$

其中，

$\tilde{\mathbf{x}}_I$ 和 $\mathbf{n}_I$ 为

$$\tilde{\mathbf{x}}_I = \begin{pmatrix} {}^I_G\tilde{\boldsymbol{\theta}}^\top & \tilde{\mathbf{b}}_g^\top & {}^G\tilde{\mathbf{v}}_I^\top & \tilde{\mathbf{b}}_a^\top & {}^G\tilde{\mathbf{p}}_I^\top & {}^I_C\tilde{\boldsymbol{\theta}}^\top & {}^I\tilde{\mathbf{p}}_C^\top \end{pmatrix}^\top \in \mathbb{R}^{21\times 1}$$

$$\mathbf{n}_I^\top = \begin{pmatrix} \mathbf{n}_g^\top & \mathbf{n}_{wg}^\top & \mathbf{n}_a^\top & \mathbf{n}_{wa}^\top \end{pmatrix}^\top \in \mathbb{R}^{12\times 1}$$

$\mathbf{F}$ 和 $\mathbf{G}$ 为（代码在 `MsckfVio::processModel`，文献 [3] 附录中 $\mathbf{G}$ 计算有误，代码正确）

$$\mathbf{F}_{21\times 21} = \begin{pmatrix} -\lfloor\hat{\boldsymbol{\omega}}_\times\rfloor & -\mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ -C\left({}^I_G\hat{\mathbf{q}}\right)^\top \lfloor\hat{\mathbf{a}}_\times\rfloor & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -C\left({}^I_G\hat{\mathbf{q}}\right)^\top & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{pmatrix}$$

$$\mathbf{G}_{21\times 12} = \begin{pmatrix} -\mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -C\left({}^I_G\hat{\mathbf{q}}\right)^\top & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_3 \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{pmatrix}$$

3.4.2 IMU 状态向量预测

代码在 `MsckfVio::predictNewState`。

- 姿态预测（0 阶四元数积分，根据 [5] 中 122 式）

$${}^I_G\mathbf{q}(t_{k+1}) = \left(\cos\left(\frac{|\omega|}{2}\Delta t\right)\cdot \mathbf{I}_{4\times 4} + \frac{1}{|\omega|}\sin\left(\frac{|\omega|}{2}\Delta t\right)\cdot\ (\omega)\right) {}^I_G\mathbf{q}(t_k) \tag{4}$$

- 位置和速度预测（4 阶 Runge-Kutta 积分）

$$y_{n+1} = y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{5}$$

3.4.3 系统状态协方差预测

代码在 `MsckfVio::processModel`。

离散时间状态转移矩阵

$$\begin{aligned}\mathbf{\Phi}_k &= \mathbf{\Phi}(t_{k+1}, t_k)\\ &= \exp\left(\int_{t_k}^{t_{k+1}} \mathbf{F}(\tau)d\tau\right)\\ &= \exp(\mathbf{F}\Delta t)\\ &\approx \mathbf{I} + \mathbf{F}\Delta t + \frac{1}{2}(\mathbf{F}\Delta t)^2 + \frac{1}{6}(\mathbf{F}\Delta t)^3 \quad (\Delta t\text{比较小时})\end{aligned} \tag{6}$$

离散时间噪声协方差矩阵

$$\begin{aligned}\mathbf{Q}_k &= \int_{t_k}^{t_{k+1}} \mathbf{\Phi}(t_{k+1}, \tau)\mathbf{G}\mathbf{Q}_I\mathbf{G}\mathbf{\Phi}(t_{k+1}, \tau)^\top d\tau\\ &\approx \mathbf{\Phi}_k\mathbf{G}\mathbf{Q}_I\mathbf{G}^T{\mathbf{\Phi}_k}^T\Delta t\end{aligned} \tag{7}$$

IMU 状态传播协方差矩阵为

$$\mathbf{P}_{II_{k+1|k}} = \mathbf{\Phi}_k\mathbf{P}_{II_{k|k}}\mathbf{\Phi}_k^\top + \mathbf{Q}_k$$

系统状态传播协方差矩阵拆解表示为

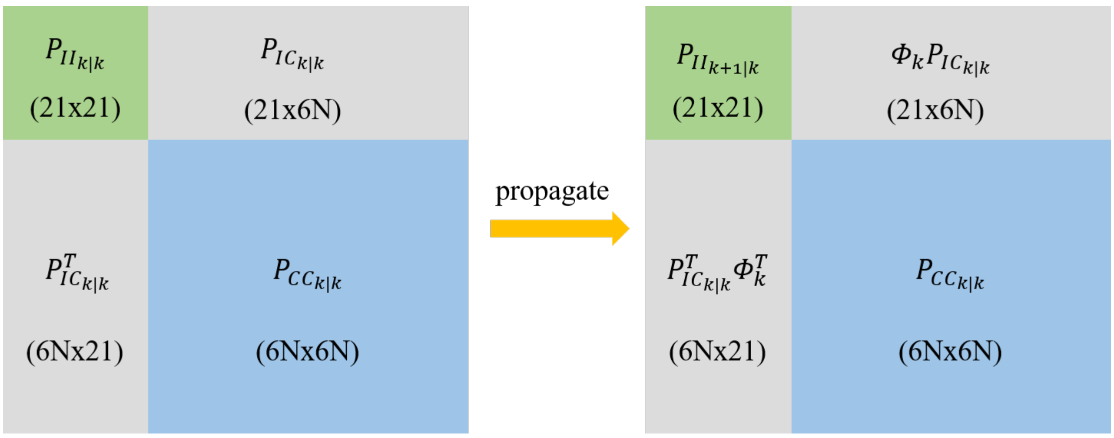
$$\mathbf{P}_{k|k} = \begin{pmatrix}\mathbf{P}_{II_{k|k}} & \mathbf{P}_{IC_{k|k}}\\ \mathbf{P}_{IC_{k|k}}^\top & \mathbf{P}_{CC_{k|k}}\end{pmatrix}$$

其传播形式表示为

$$\mathbf{P}_{k+1|k} = \begin{pmatrix}\mathbf{P}_{II_{k+1|k}} & \mathbf{\Phi}_k\mathbf{P}_{IC_{k|k}}\\ \mathbf{P}_{IC_{k|k}}^\top\mathbf{\Phi}_k^\top & \mathbf{P}_{CC_{k|k}}\end{pmatrix}$$

其中 Camera 的 covariance 暂时还没有变化是因为这个时间段图像还没有到来，只有 IMU 的影响，但是会影响到 IMU 与 Camera 协方差。

整个状态 (IMU+Camera) 的 covariance 传播过程如图所示 [2]：



3.5 State Augmentation

代码在 `MsckfVio::stateAugmentation`。

3.5.1 相机状态向量扩增

根据 IMU 位姿求取 Camera 位姿

$$
\begin{aligned}
{}^C_G\hat{\mathbf{q}} &= {}^C_I\hat{\mathbf{q}} \otimes {}^I_G\hat{\mathbf{q}} \\
{}^G\hat{\mathbf{p}}_C &= {}^G\hat{\mathbf{p}}_C + C\left({}^I_G\hat{\mathbf{q}}\right)^\top {}^I\hat{\mathbf{p}}_C
\end{aligned}
\tag{8}
$$

3.5.2 状态协方差扩增

增广的状态协方差矩阵为（为什么）

$$
\begin{aligned}
\mathbf{P}'_{k|k} &= \begin{pmatrix} \mathbf{I}_{21+6N} \\ \mathbf{J} \end{pmatrix} \mathbf{P}_{k|k} \begin{pmatrix} \mathbf{I}_{21+6N} \\ \mathbf{J} \end{pmatrix}^\top \\
&= \begin{bmatrix} \mathbf{P}_{k|k} & (\mathbf{J}\mathbf{P}_{k|k})^T \\ \mathbf{J}\mathbf{P}_{k|k} & \mathbf{J}\mathbf{P}_{k|k}\mathbf{J}^T \end{bmatrix}
\end{aligned}
\tag{9}
$$

其中（下式参考 [3]，而代码参考 [1]，哪一个正确），

$$
\mathbf{J} = \frac{\partial \tilde{\mathbf{x}}_{C_i}}{\partial \tilde{\mathbf{x}}} = \frac{\partial\left({}^{C_i}_G\tilde{\boldsymbol{\theta}}^\top \quad {}^G\tilde{\mathbf{p}}_{C_i}^\top\right)^\top}{\partial\left(\tilde{\mathbf{x}}_I^\top \quad \tilde{\mathbf{x}}_{C_1}^\top \quad \cdots \quad \tilde{\mathbf{x}}_{C_N}^\top\right)^\top} = \begin{pmatrix} \mathbf{J}_I & \mathbf{0}_{6\times 6N} \end{pmatrix} \in \mathbb{R}^{6\times(21+6N)}
$$

$$
\mathbf{J}_I = \frac{\partial \tilde{\mathbf{x}}_{C_i}}{\partial \tilde{\mathbf{x}}_I} = \begin{pmatrix} C\left({}^I_G\hat{\mathbf{q}}\right) & \mathbf{0}_{3\times 9} & \mathbf{0}_{3\times 3} & \mathbf{I}_3 & \mathbf{0}_{3\times 3} \\ -C\left({}^I_G\hat{\mathbf{q}}\right)^\top \lfloor {}^I\hat{\mathbf{p}}_{C\times} \rfloor & \mathbf{0}_{3\times 9} & \mathbf{I}_3 & \mathbf{0}_{3\times 3} & \mathbf{I}_3 \end{pmatrix} \in \mathbb{R}^{6\times 21}
$$

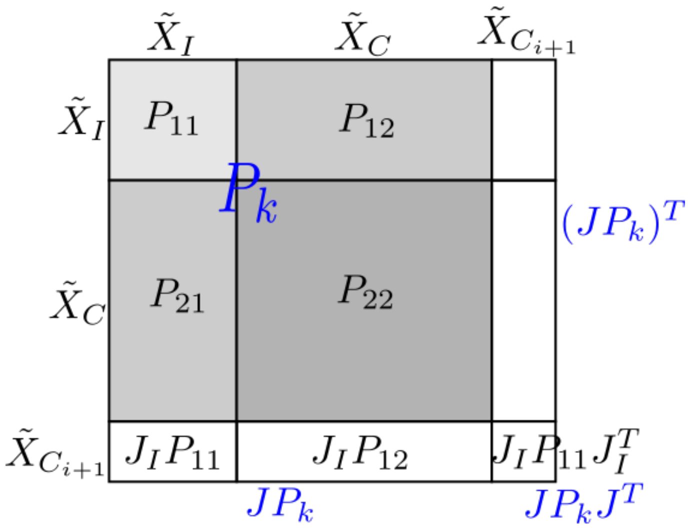
为表示方便，拆解 $\mathbf{P}_{k|k}$ 为

$$
\mathbf{P}_{k|k} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{bmatrix}
\tag{10}
$$

则

$$
\mathbf{P}'_{k|k} = \begin{bmatrix} \mathbf{P}_{k|k} & \mathbf{J}_P^T \\ \mathbf{J}_P & \mathbf{J}_I\mathbf{P}_{11}\mathbf{J}_I^T \end{bmatrix} \quad \text{where} \quad \mathbf{J}_P = \begin{pmatrix} \mathbf{J}_I\mathbf{P}_{11} & \mathbf{J}_I\mathbf{P}_{12} \end{pmatrix}
\tag{11}
$$

图形化表示如下



3.6 Measurement Update

3.6.1 Measurement Model

1. 计算获取世界坐标系下的 3D 特征点，代码在 `Feature::generateInitialGuess`
   - 根据对极几何原理，三角化计算初始深度，得到初始三维点坐标
   - 通过 L-M 算法迭代优化得到更加精确的世界系三维点

2. 投影世界系三维点到相机系

$$
\begin{aligned}
{}^{C_{i,1}}\mathbf{p}_j &= \begin{pmatrix} {}^{C_{i,1}}X_j \\ {}^{C_{i,1}}Y_j \\ {}^{C_{i,1}}Z_j \end{pmatrix} = C\left({}^{C_{i,1}}_{G}\mathbf{q}\right)\left({}^{G}\mathbf{p}_j - {}^{G}\mathbf{p}_{C_{i,1}}\right) \\
{}^{C_{i,2}}\mathbf{p}_j &= \begin{pmatrix} {}^{C_{i,2}}X_j \\ {}^{C_{i,2}}Y_j \\ {}^{C_{i,2}}Z_j \end{pmatrix} = C\left({}^{C_{i,2}}_{G}\mathbf{q}\right)\left({}^{G}\mathbf{p}_j - {}^{G}\mathbf{p}_{C_{i,2}}\right) \\
&= C\left({}^{C_{i,2}}_{C_{i,1}}\mathbf{q}\right)\left({}^{C_{i,1}}\mathbf{p}_j - {}^{C_{i,1}}\mathbf{p}_{C_{i,2}}\right)
\end{aligned}
$$

3. 线性化测量模型，视觉测量残差可近似表示为

$$
\mathbf{r}_i^j = \mathbf{z}_i^j - \hat{\mathbf{z}}_i^j = \mathbf{H}_{C_i}^j \tilde{\mathbf{x}}_{C_i} + \mathbf{H}_{f_i}^j {}^{G}\tilde{\mathbf{p}}_j + \mathbf{n}_i^j \in \mathbb{R}^{4\times 1} \tag{12}
$$

   其中，测量雅克比矩阵（代码在 `MsckfVio::measurementJacobian`）

$$
\begin{aligned}
\mathbf{H}_{C_i}^j &= \frac{\partial \mathbf{z}_i^j}{\partial {}^{C_{i,1}}\mathbf{p}_j} \cdot \frac{\partial {}^{C_{i,1}}\mathbf{p}_j}{\partial \mathbf{x}_{C_{i,1}}} + \frac{\partial \mathbf{z}_i^j}{\partial {}^{C_{i,2}}\mathbf{p}_j} \cdot \frac{\partial {}^{C_{i,2}}\mathbf{p}_j}{\partial \mathbf{x}_{C_{i,1}}} \in \mathbb{R}^{4\times 6} \\
\mathbf{H}_{f_i}^j &= \frac{\partial \mathbf{z}_i^j}{\partial {}^{C_{i,1}}\mathbf{p}_j} \cdot \frac{\partial {}^{C_{i,1}}\mathbf{p}_j}{\partial {}^{G}\mathbf{p}_j} + \frac{\partial \mathbf{z}_i^j}{\partial {}^{C_{i,2}}\mathbf{p}_j} \cdot \frac{\partial {}^{C_{i,2}}\mathbf{p}_j}{\partial {}^{G}\mathbf{p}_j} \in \mathbb{R}^{4\times 3}
\end{aligned} \tag{13}
$$

   问题：Modifty the measurement Jacobian to ensure observability constrain

4. 叠加对同一特征点的多个观测（代码在 `MsckfVio::featureJacobian`）

$$
\mathbf{r}^j = \mathbf{H}_{\mathbf{x}}^j \tilde{\mathbf{x}} + \mathbf{H}_f^j {}^{G}\tilde{\mathbf{p}}_j + \mathbf{n}^j \quad \text{with} \quad \mathbf{H}_{\mathbf{x}}^j \in \mathbb{R}^{4M\times 6}, \mathbf{H}_f^j \in \mathbb{R}^{4M\times 3}
$$

5. 为避免 ${}^{G}\mathbf{p}_j$ 的不确定性对测量残差的影响，将残差投影到 $\mathbf{H}_{f_i}^j$ 的左零空间

$$
\mathbf{r}_o^j = \mathbf{V}^\top \mathbf{r}^j = \mathbf{V}^\top \mathbf{H}_{\mathbf{x}}^j \tilde{\mathbf{x}} + \mathbf{V}^\top \mathbf{n}^j = \mathbf{H}_{\mathbf{x},o}^j \tilde{\mathbf{x}} + \mathbf{n}_o^j \tag{14}
$$

   零空间 $\mathbf{V}$ 通过 SVD 分解得到（代码在 `MsckfVio::featureJacobian`）

```
// Project the residual and Jacobians onto the nullspace of H_fj.
JacobiSVD<MatrixXd> svd_helper(H_fj, ComputeFullU | ComputeThinV);
MatrixXd A = svd_helper.matrixU().rightCols(jacobian_row_size - 3);
H_x = A.transpose() * H_xj;
r   = A.transpose() * r_j;
```

6. 叠加所有特征点的多个观测，得到

$$
\mathbf{r}_o = \mathbf{H}_{\mathbf{X}} \widetilde{\mathbf{X}} + \mathbf{n}_o \tag{15}
$$

```
if (gatingTest(H_xj, r_j, cam_state_ids.size()-1)) {
  H_x.block(stack_cntr, 0, H_xj.rows(), H_xj.cols()) = H_xj;
  r.segment(stack_cntr, r_j.rows()) = r_j;
  stack_cntr += H_xj.rows();
}
```

3.6.2 能观性约束

OC-EKF

3.6.3 EKF Update

代码在 `MsckfVio::measurementUpdate`。
根据 [1]，为降低 EKF 更新的计算复杂度，对 $\mathbf{H_X}$ 进行 QR 分解

$$\mathbf{H_X} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{bmatrix} \begin{bmatrix} \mathbf{T}_H \\ \mathbf{0} \end{bmatrix} \tag{16}$$

带入式 (15)，得

$$\mathbf{r}_o = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{bmatrix} \begin{bmatrix} \mathbf{T}_H \\ \mathbf{0} \end{bmatrix} \tilde{\mathbf{X}} + \mathbf{n}_o \tag{17}$$

$$\begin{bmatrix} \mathbf{Q}_1^T \mathbf{r}_o \\ \mathbf{Q}_2^T \mathbf{r}_o \end{bmatrix} = \begin{bmatrix} \mathbf{T}_H \\ \mathbf{0} \end{bmatrix} \tilde{\mathbf{X}} + \begin{bmatrix} \mathbf{Q}_1^T \mathbf{n}_o \\ \mathbf{Q}_2^T \mathbf{n}_o \end{bmatrix} \tag{18}$$

上式 $\mathbf{Q}_2^T \mathbf{r}_o$ 仅含有噪声，对其忽略，取 $\mathbf{Q}_1^T \mathbf{r}_o$ 用于 EKF 更新

$$\mathbf{r}_n = \mathbf{Q}_1^T \mathbf{r}_o = \mathbf{T}_H \widetilde{\mathbf{X}} + \mathbf{n}_n \tag{19}$$

噪声向量 $\mathbf{n}_n$ 的协方差矩阵为

$$\mathbf{R}_n = \mathbf{Q}_1^T \mathbf{R}_o \mathbf{Q}_1 = \sigma_{\mathrm{im}}^2 \mathbf{I}_r \tag{20}$$

计算 Kalman 增益

$$\mathbf{K} = \mathbf{P} \mathbf{T}_H^T \left(\mathbf{T}_H \mathbf{P} \mathbf{T}_H^T + \mathbf{R}_n \right)^{-1} \tag{21}$$

更新系统误差状态

$$\Delta \mathbf{X} = \mathbf{K} \mathbf{r}_n \tag{22}$$

更新系统状态

$$\mathbf{X}_{k+1} = \mathbf{X}_k \oplus \Delta \mathbf{X} \tag{23}$$

更新系统状态协方差矩阵

$$\mathbf{P}_{k+1|k+1} = \left(\mathbf{I}_\xi - \mathbf{K} \mathbf{T}_H \right) \mathbf{P}_{k+1|k} \left(\mathbf{I}_\xi - \mathbf{K} \mathbf{T}_H \right)^T + \mathbf{K} \mathbf{R}_n \mathbf{K}^T \tag{24}$$

参考文献

[1] Anastasios I Mourikis and Stergios I Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572. IEEE, 2007.

[2] 钟心亮. 一步步深入了解 s-msckf. http://www.xinliang-zhong.vip/msckf_notes/, 2019.

[3] Ke Sun, Kartik Mohta, Bernd Pfrommer, Michael Watterson, Sikang Liu, Yash Mulgaonkar, Camillo J Taylor, and Vijay Kumar. Robust stereo visual inertial odometry for fast autonomous flight. *IEEE Robotics and Automation Letters*, 3(2):965–972, 2018.

[4] Joan Solà. Quaternion kinematics for the error-state kalman filter. *CoRR*, abs/1711.02508, 2017.

[5] Nikolas Trawny and Stergios I Roumeliotis. Indirect kalman filter for 3d attitude estimation. *University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep*, 2:2005, 2005.