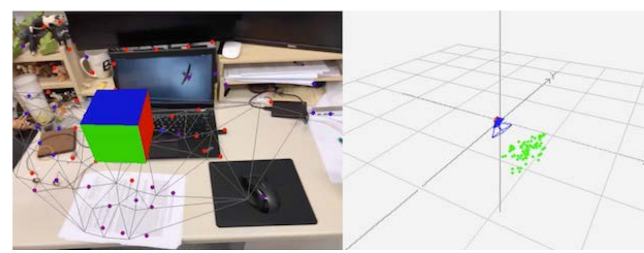
Monocular Visual-Inertial SLAM

Shaojie Shen
Assistant Professor, HKUST
Director, HKUST-DJI Joint Innovation Laboratory

Why Monocular?

- Minimum structural requirements
- Widely available sensors
- Applications:
  - State estimation for small drones
  - Mobile augmented reality

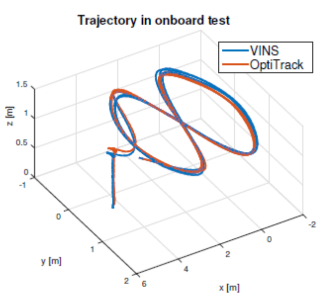


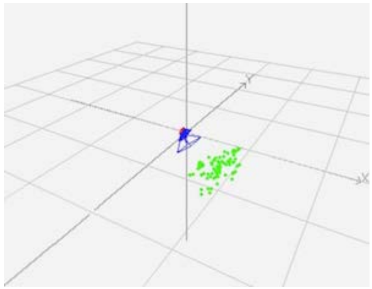
Why IMU?

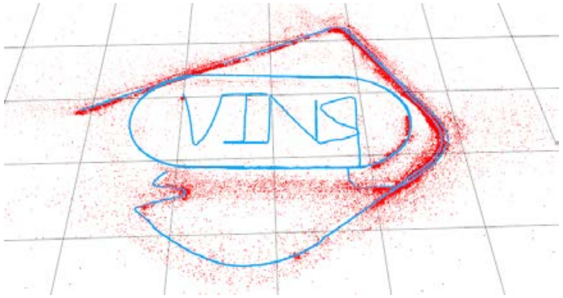
- IMU measures:
  - Linear acceleration
  - Angular velocity
- Pros:
  - Almost always available and outlier-free
  - Very high-rate measurements
  - Very mature technology, widely available at very low cost
  - Remarkable performance improvement during aggressive motions
- Cons:
  - Noisy sensor, cannot double integrate to obtain position
  - Synchronization and inter-sensor calibration requirements
  - Observability and numerical stability issues
  - Unable to operate when inertial and visual measurements are not in the same frame (e.g. on cars or trains)

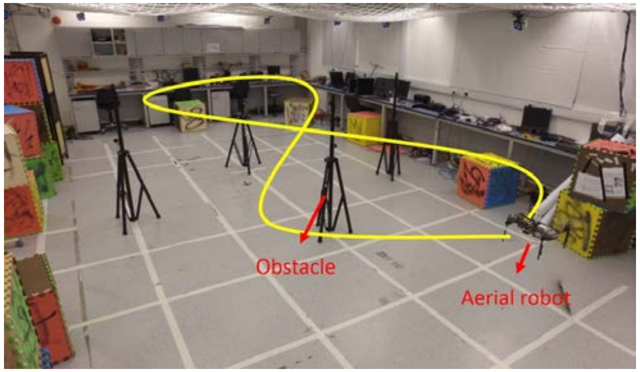
Requirements

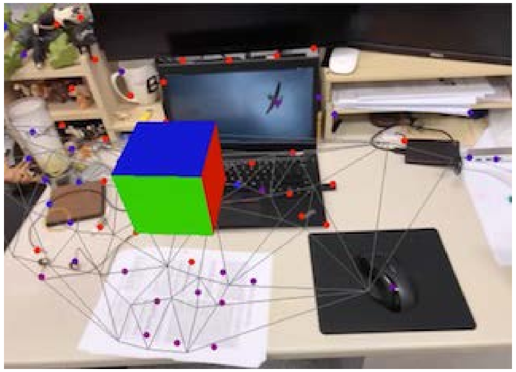
- Metric scale estimation using only one camera
- Mostly for state estimation (localization), map is sparse
- Robust and smooth odometry – local accuracy
- Loop closure – global consistency











Related work

- MSC-KF (Mourikis and Roumeliotis, 2007)
- OKVIS (Leutenegger, et al., 2015)
  - Code: https://github.com/ethz-asl/okvis
- Visual-Inertial ORB SLAM (Mur-Artal and Tardos, 2017)
  - No official source code available yet
- Apple ARKit
- Google ARCore

Our Solution: VINS-Mono

HKUST-Aerial-Robotics / **VINS-Mono**

Unwatch 116 | Unstar 724 | Fork 477

Code | Issues 114 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

A Robust and Versatile Monocular Visual-Inertial State Estimator

Edit

state-estimation | vio | vins | Manage topics

58 commits | 3 branches | 0 releases | 4 contributors | GPL-3.0

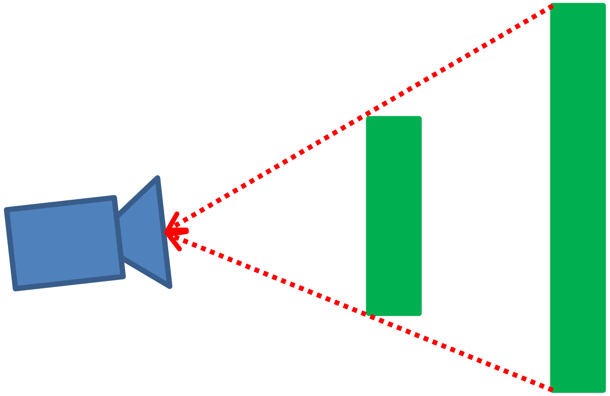
Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

qintonguav Update README.md — Latest commit 65b4390 17 days ago

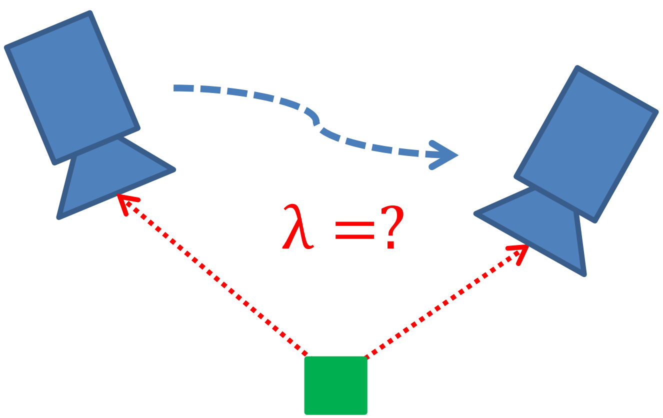
ar_demo	modify eigen3 in cmake	3 months ago
benchmark_publisher	another warning	3 months ago
camera_model	add Eigen3 cmake	3 months ago
config	add realsense config; avoid imu disorder; fix reloclization visualiza…	3 months ago
feature_tracker	add realsense config; avoid imu disorder; fix reloclization visualiza…	3 months ago
pose_graph	add realsense config; avoid imu disorder; fix reloclization visualiza…	3 months ago
support_files	2017-12-29 New features: Add map merge, pose graph reuse, online temp…	3 months ago
vins_estimator	add realsense config; avoid imu disorder; fix reloclization visualiza…	3 months ago

Challenges: Monocular Vision

- Scale ambiguity



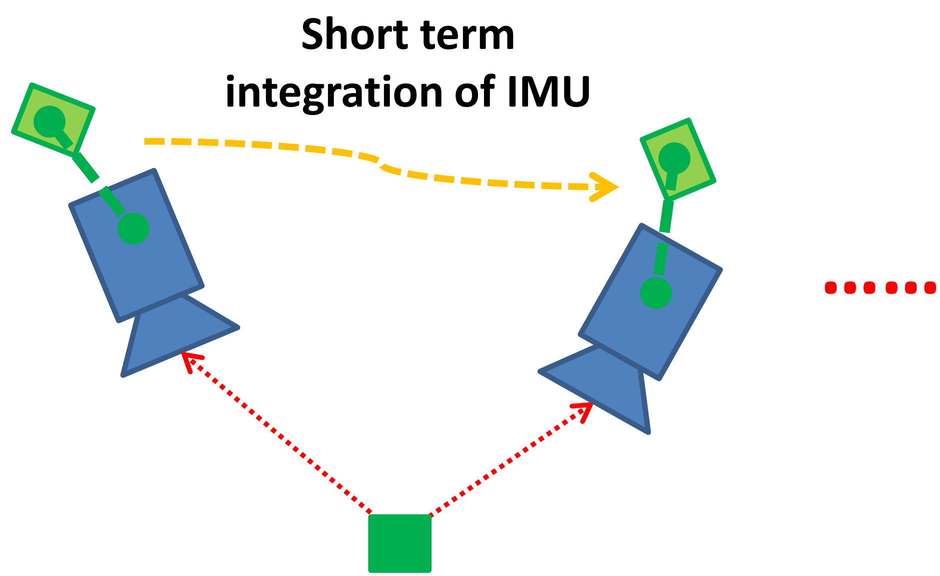
- Up-to-scale motion estimation and 3D reconstruction (Structure from Motion)



Challenges: Monocular Visual-Inertial Systems

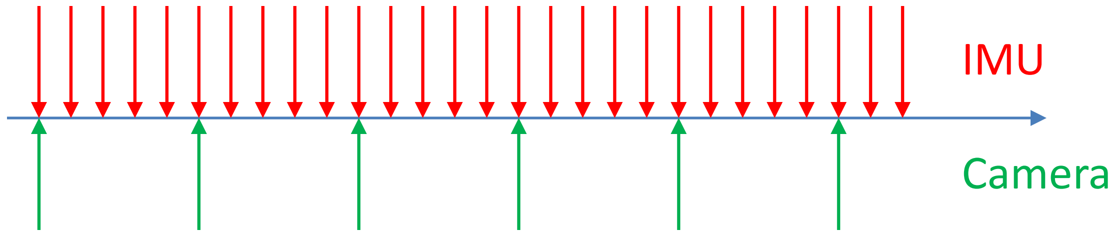
- With IMU, scale is observable (via accelerometer), but…
  - Requires recovery of initial velocity and attitude (gravity)
  - Requires online calibration camera-IMU extrinsic parameters
  - Requires multi-observation constraints

$$\mathrm{v}_0 = ?$$
$$\mathrm{g}_0 = ?$$
$$\mathrm{R}_\mathrm{c}^\mathrm{b} = ?$$
$$\mathrm{p}_\mathrm{c}^\mathrm{b} = ?$$

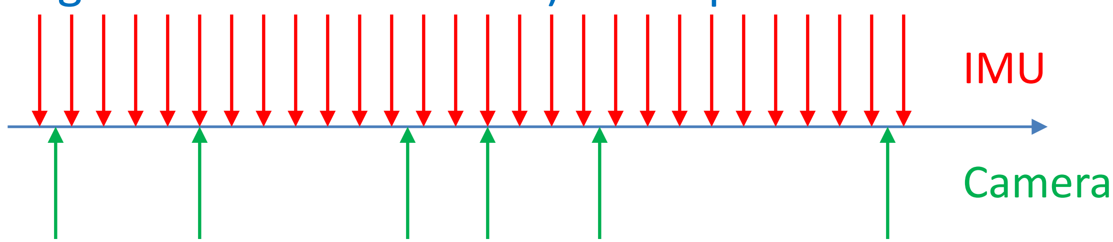


Challenges: Synchronization

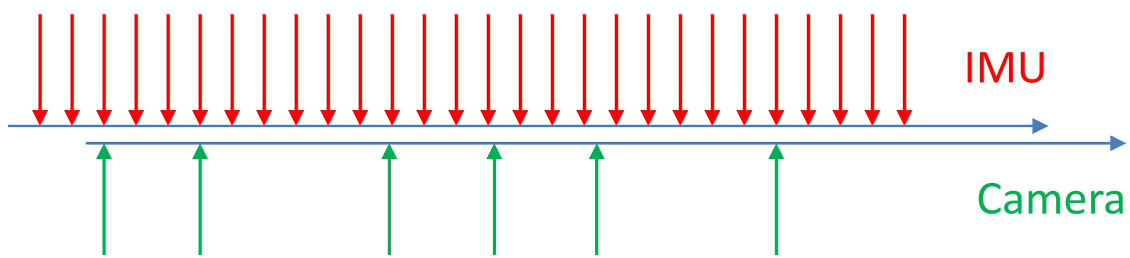
- Best: Sensors are hardware-triggered



- OK: Sensors have the same clock (e.g. running on the same system clock or have global clock correction) but capture data at different times

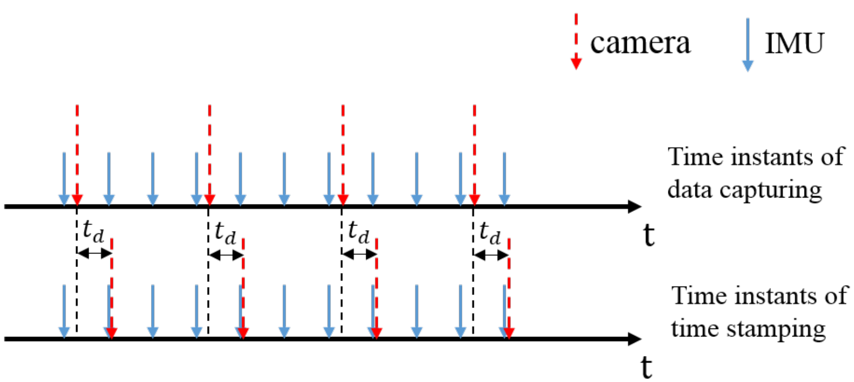


- Bad: Sensors have different clocks (e.g. each sensor has its own oscillator)

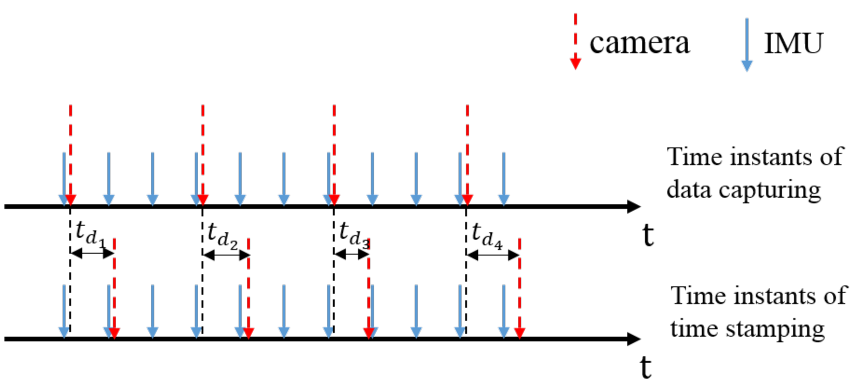


Challenges: Timestamps

- Timestamp: how the time for each sensor measurement is tagged
- Best: timestamping is done at data capture
- OK: fixed latency for time stamping
  - e.g. time is tagged on low-level hardware after some fixed-duration data processing, and will not be affected by any dynamic OS scheduling tasks
- Bad: variable latency in time stamping
  - e.g. plug two sensors into USB ports and time stamp according to the PC time. Time stamping is affected by data transmission latency from the sensor to PC



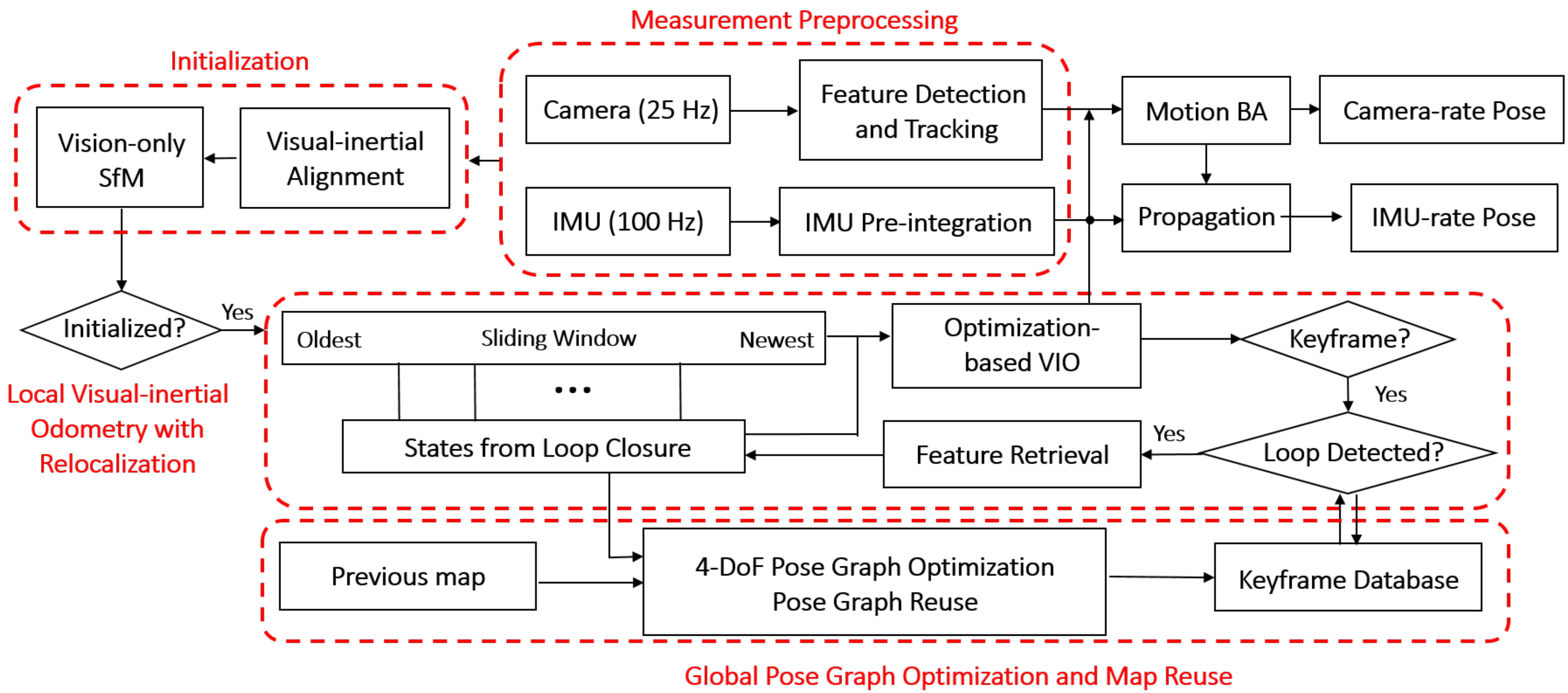
Good



Bad

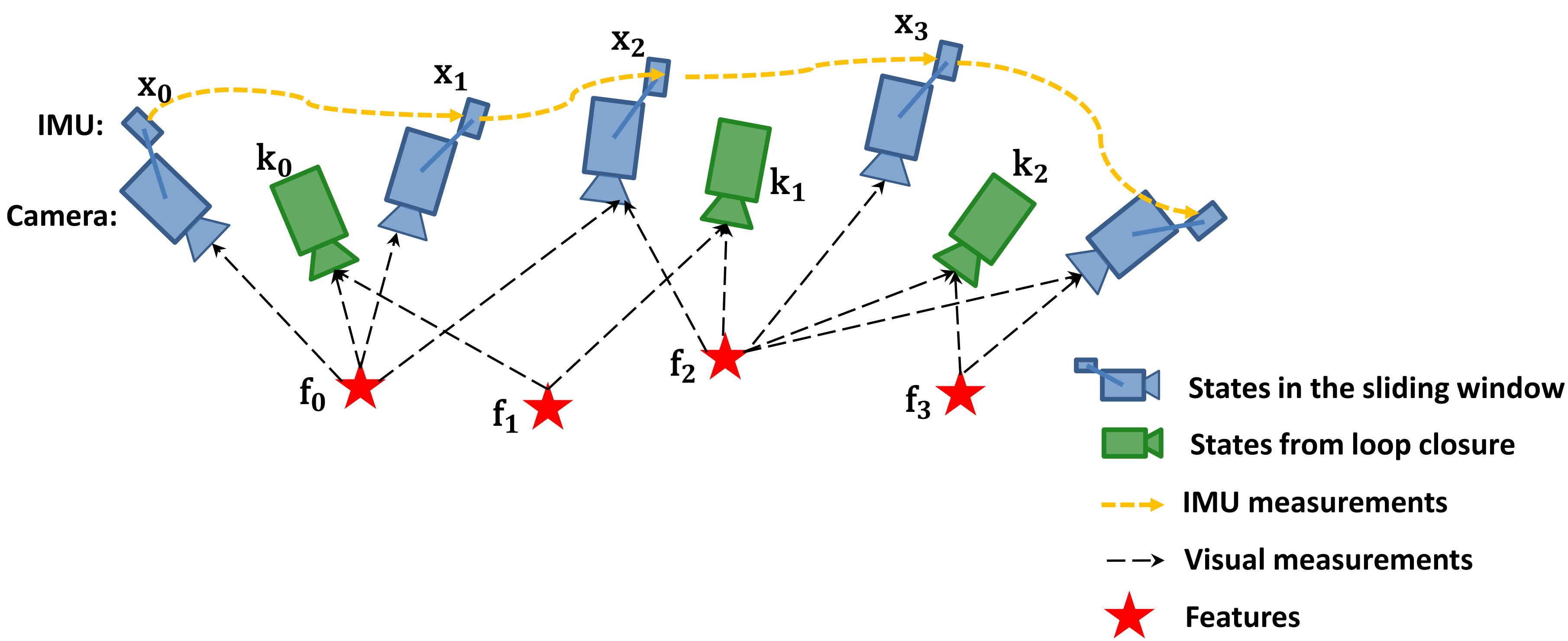
Monocular Visual-Inertial SLAM

- System diagram



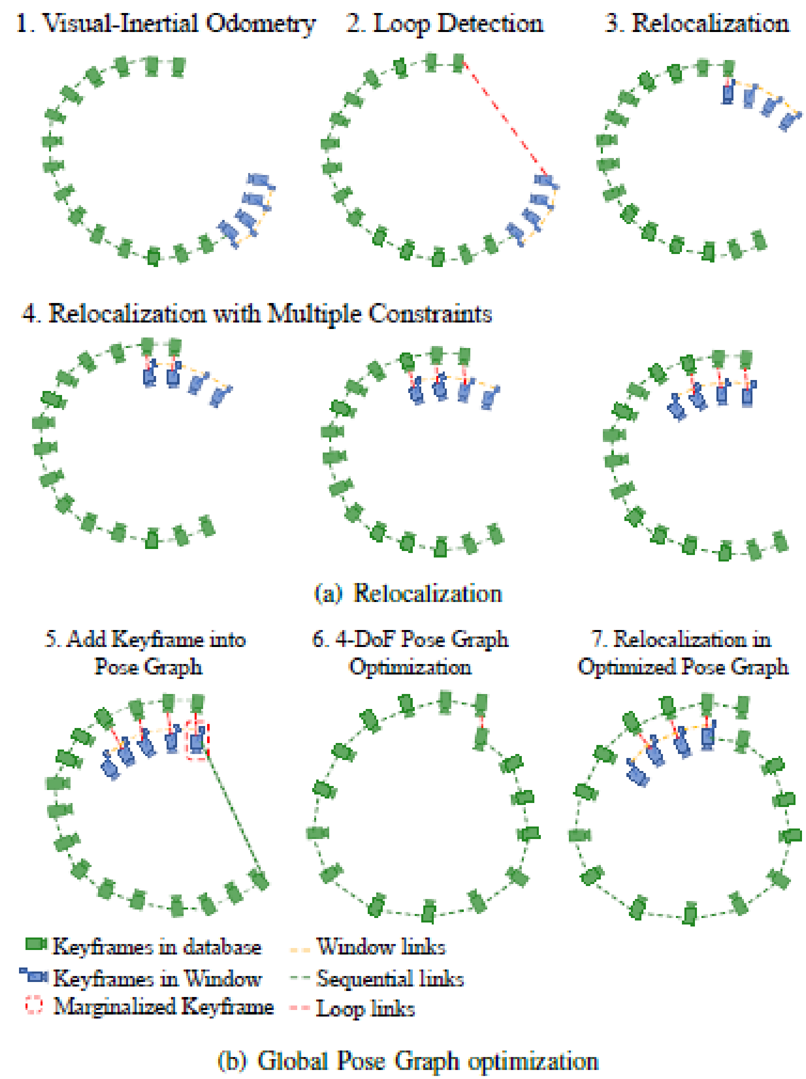
Monocular Visual-Inertial SLAM

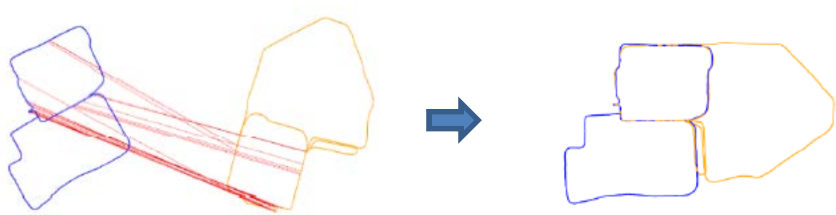
- Monocular visual-inertial odometry with relocalization
  - For local accuracy
  - Achieved via sliding window visual-inertial bundle adjustment



Monocular Visual-Inertial SLAM

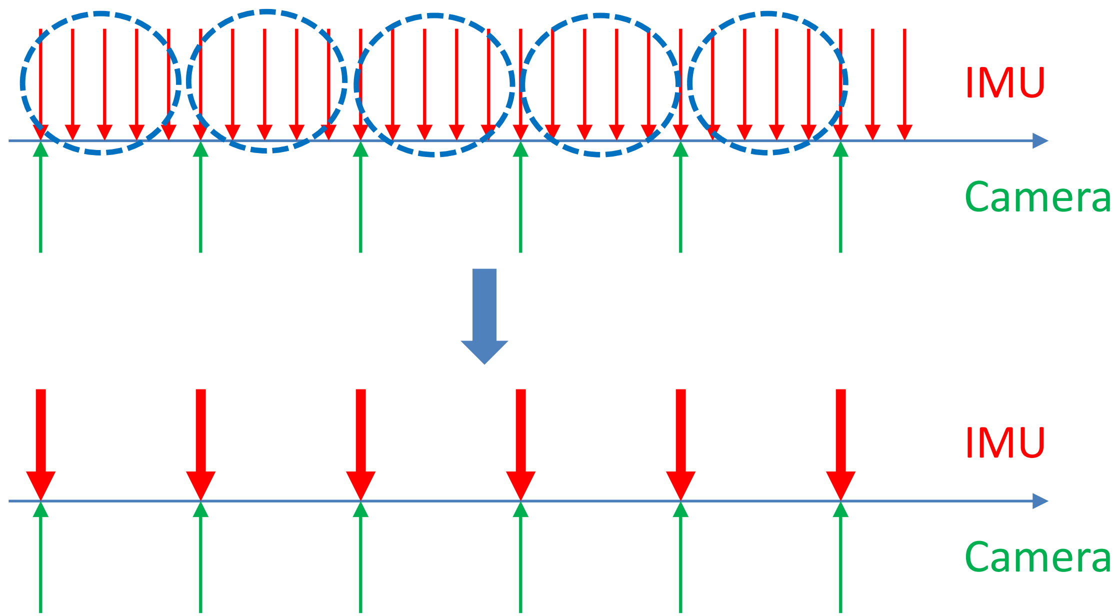
- Global pose graph SLAM
  - For global consistency
  - Fully integrated with tightly-coupled re-localization
- Map reuse
  - Save map at any time
  - Load map and re-localize with respect to it
  - Pose graph merging





How to Use IMU?

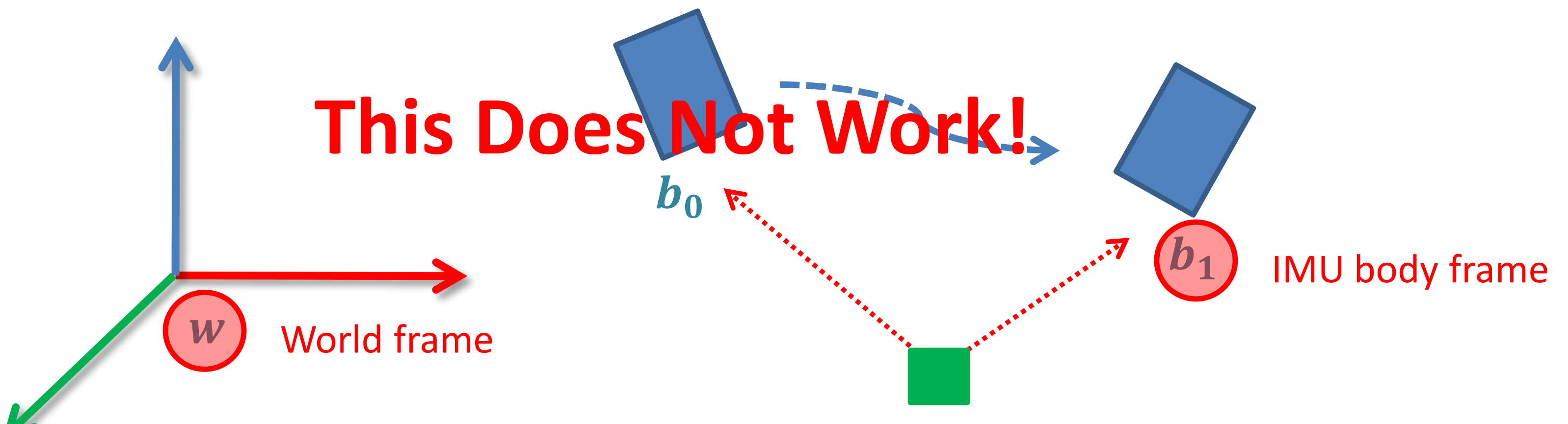
- IMU integration
  - IMU has higher rate than camera
  - Cannot estimate all IMU states
  - Need to integration IMU measurements



The Bad of IMU Integration in the Global Frame

- IMU integration in world frame
  - Requires global rotation at the time of integration

$$
\begin{aligned}
\mathbf{p}^w_{b_{k+1}} &= \mathbf{p}^w_{b_k} + \mathbf{v}^w_{b_k}\Delta t_k \\
&\quad + \iint_{t\in[t_k,t_{k+1}]} \left(\mathbf{R}^w_t(\hat{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a) - \mathbf{g}^w\right) dt^2 \\
\mathbf{v}^w_{b_{k+1}} &= \mathbf{v}^w_{b_k} + \int_{t\in[t_k,t_{k+1}]} \left(\mathbf{R}^w_t(\hat{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a) - \mathbf{g}^w\right) dt \\
\mathbf{q}^w_{b_{k+1}} &= \mathbf{q}^w_{b_k} \otimes \int_{t\in[t_k,t_{k+1}]} \frac{1}{2}\boldsymbol{\Omega}(\hat{\boldsymbol{\omega}}_t - \mathbf{b}_{w_t} - \mathbf{n}_w)\mathbf{q}^{b_k}_t dt, \quad \boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -\lfloor\boldsymbol{\omega}\rfloor_\times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^T & 0 \end{bmatrix}, \lfloor\boldsymbol{\omega}\rfloor_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}.
\end{aligned}
$$

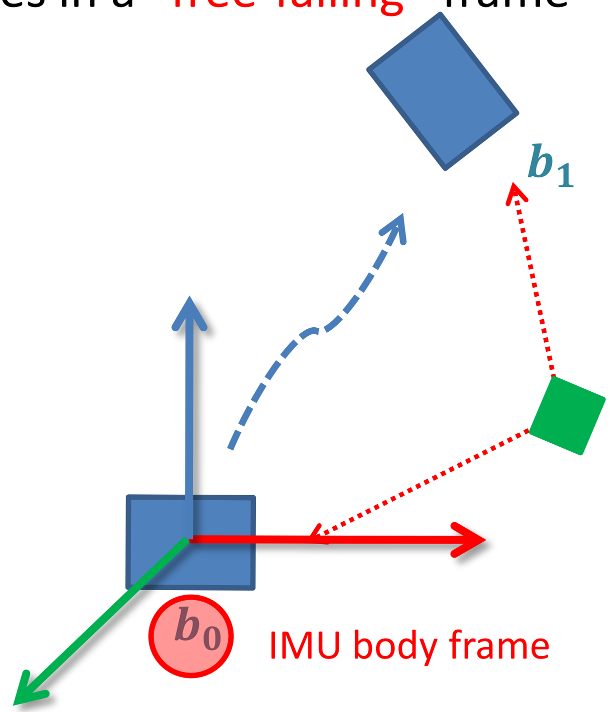


IMU Pre-Integration on Manifold

- IMU integration in the body frame of first pose of interests
  - IMU Integration without initialization
  - Can use any discrete implementation for numerical integration
  - Intuitive: “position” and “velocity” changes in a “free-falling” frame

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

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



IMU Pre-Integration on Manifold

- Uncertainty propagation on manifold
  - Derive the error state model for the IMU pre-integration dynamics

$$
\begin{bmatrix} \delta\dot{\boldsymbol{\alpha}}_t^{b_k} \\ \delta\dot{\boldsymbol{\beta}}_t^{b_k} \\ \delta\dot{\boldsymbol{\theta}}_t^{b_k} \\ \delta\dot{\mathbf{b}}_{a_t} \\ \delta\dot{\mathbf{b}}_{w_t} \end{bmatrix}
=
\begin{bmatrix} 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & -\mathbf{R}_t^{b_k}\lfloor \hat{\mathbf{a}}_t - \mathbf{b}_{a_t} \rfloor_\times & -\mathbf{R}_t^{b_k} & 0 \\ 0 & 0 & -\lfloor \hat{\boldsymbol{\omega}}_t - \mathbf{b}_{w_t} \rfloor_\times & 0 & -\mathbf{I} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\begin{bmatrix} \delta\boldsymbol{\alpha}_t^{b_k} \\ \delta\boldsymbol{\beta}_t^{b_k} \\ \delta\boldsymbol{\theta}_t^{b_k} \\ \delta\mathbf{b}_{a_t} \\ \delta\mathbf{b}_{w_t} \end{bmatrix}
$$

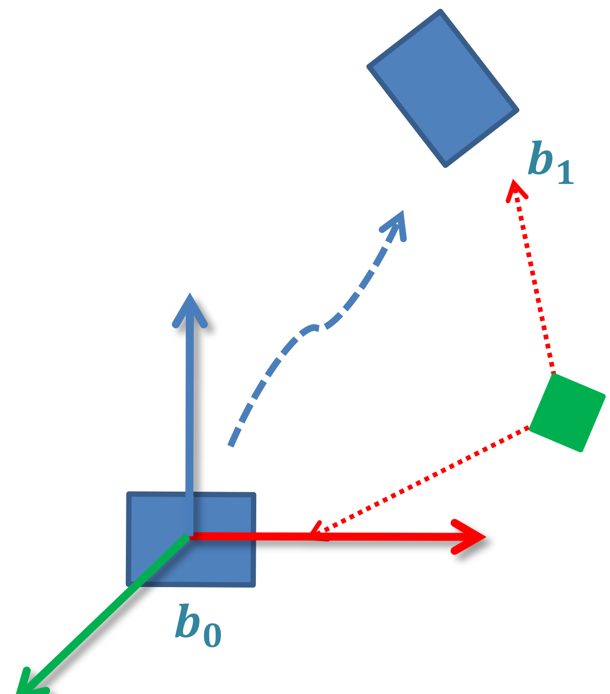
Bias uncertainty

$$
+ \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\mathbf{R}_t^{b_k} & 0 & 0 & 0 \\ 0 & -\mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{bmatrix}
\begin{bmatrix} \mathbf{n}_a \\ \mathbf{n}_w \\ \mathbf{n}_{b_a} \\ \mathbf{n}_{b_w} \end{bmatrix}
= \mathbf{F}_t \delta\mathbf{z}_t^{b_k} + \mathbf{G}_t \mathbf{n}_t .
$$

  - Discrete-time implementation

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

Covariance matrix for pre-integrated IMU measurements



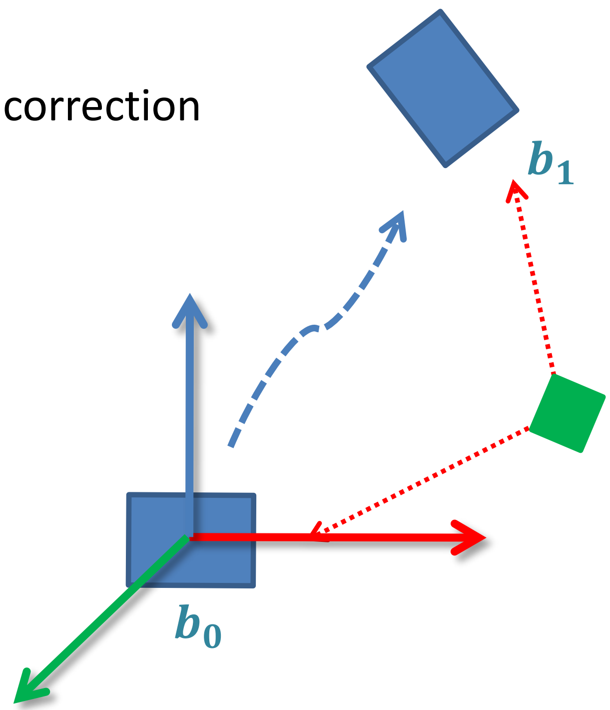
IMU Pre-Integration on Manifold

- Jacobian matrices for bias correction
  - Also derive the Jacobian of the pre-integrated measurements w.r.t. IMU bias

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

  - And write down the linearized model for bias correction

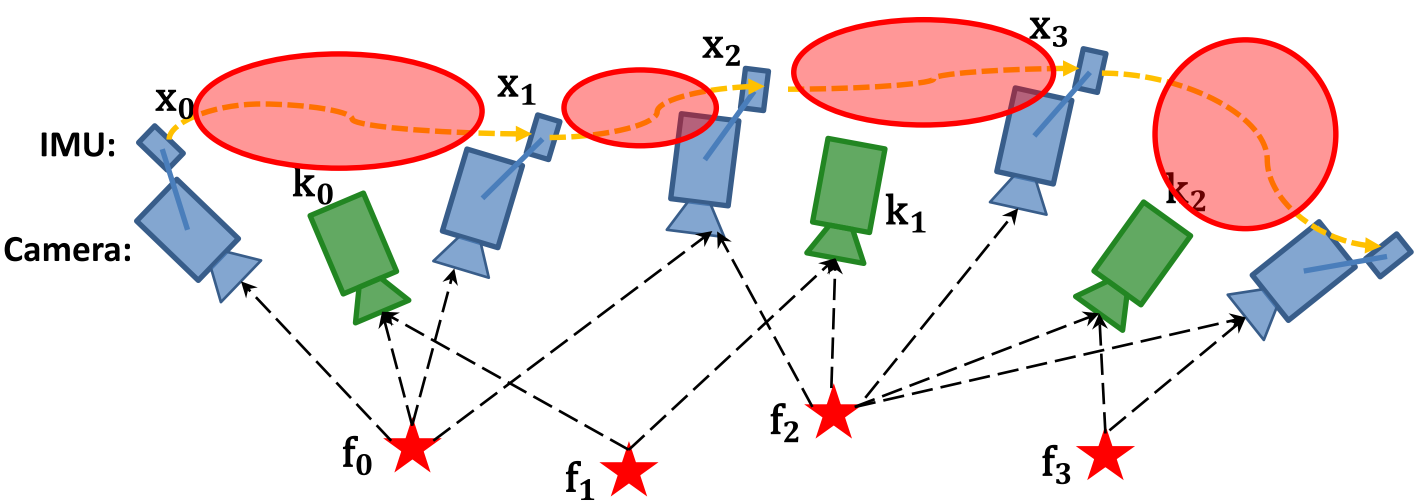
$$\alpha^{b_k}_{b_{k+1}} \approx \hat{\alpha}^{b_k}_{b_{k+1}} + \mathbf{J}^{\alpha}_{b_a} \delta \mathbf{b}_{a_k} + \mathbf{J}^{\alpha}_{b_w} \delta \mathbf{b}_{w_k}$$
$$\beta^{b_k}_{b_{k+1}} \approx \hat{\beta}^{b_k}_{b_{k+1}} + \mathbf{J}^{\beta}_{b_a} \delta \mathbf{b}_{a_k} + \mathbf{J}^{\beta}_{b_w}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



IMU Pre-Integration on Manifold

- Pre-integrated IMU measurement model
  - Describes the spatial and uncertainty relations between two states in the local sliding window

$$\begin{bmatrix} \hat{\alpha}^{b_k}_{b_{k+1}} \\ \hat{\beta}^{b_k}_{b_{k+1}} \\ \hat{\gamma}^{b_k}_{b_{k+1}} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{b_k}_{w}(\mathbf{p}^{w}_{b_{k+1}} - \mathbf{p}^{w}_{b_k} + \frac{1}{2}\mathbf{g}^{w}\Delta t_k^2 - \mathbf{v}^{w}_{b_k}\Delta t_k) \\ \mathbf{R}^{b_k}_{w}(\mathbf{v}^{w}_{b_{k+1}} + \mathbf{g}^{w}\Delta t_k - \mathbf{v}^{w}_{b_k}) \\ \mathbf{q}^{w^{-1}}_{b_k} \otimes \mathbf{q}^{w}_{b_{k+1}} \\ \mathbf{b}_{ab_{k+1}} - \mathbf{b}_{ab_k} \\ \mathbf{b}_{wb_{k+1}} - \mathbf{b}_{wb_k} \end{bmatrix}$$

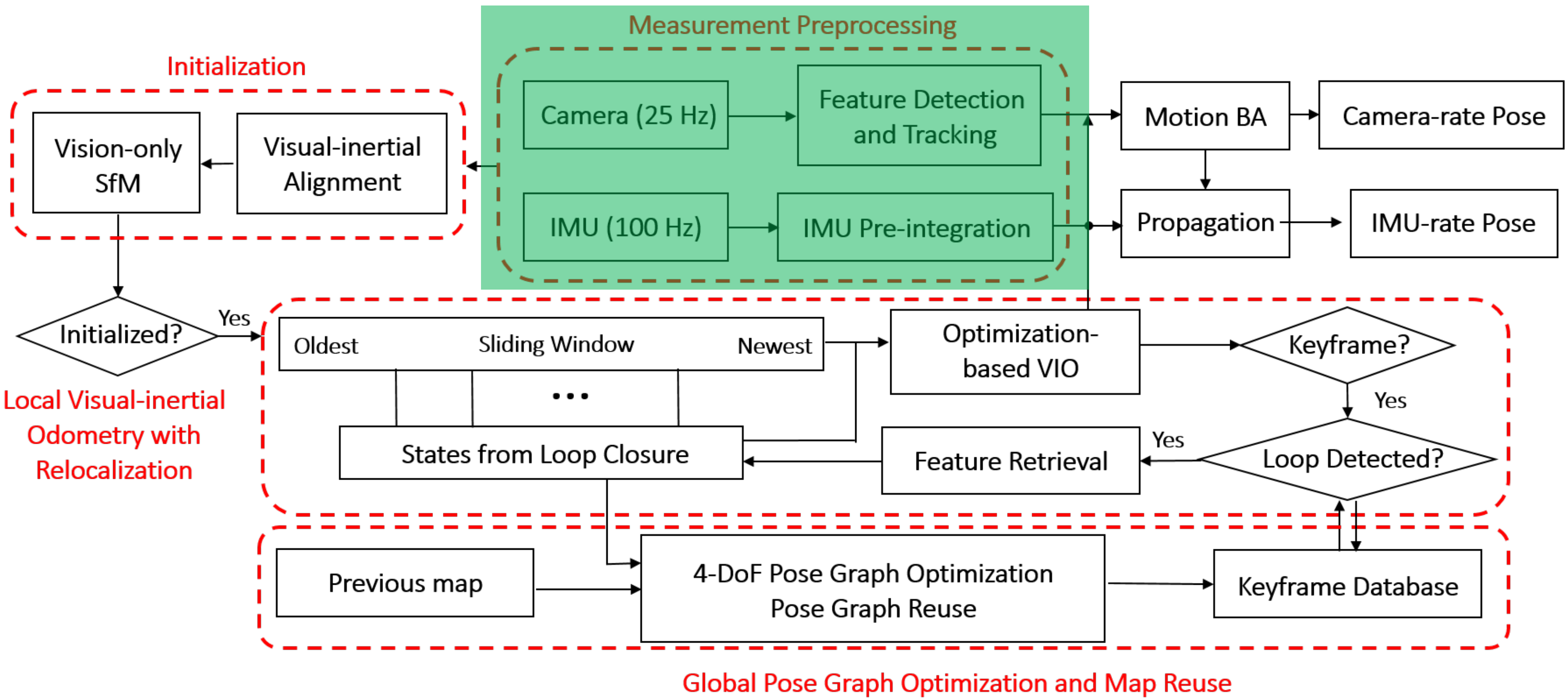


Vision Front-End

- Simple feature processing pipeline
  - Harris corners…
  - KLT tracker...
  - Track between consecutive frames
  - RANSAC for preliminary outlier removal
- Keyframe selection
  - Case 1: Rotation-compensated average feature parallax is larger than a threshold
    - Avoid numerical issues caused by poorly triangulated features
  - Case 2: Number of tracked features in the current frame is less than a threshold
    - Avoid losing tracking
  - All frames are used for optimization, but non-keyframes are removed first

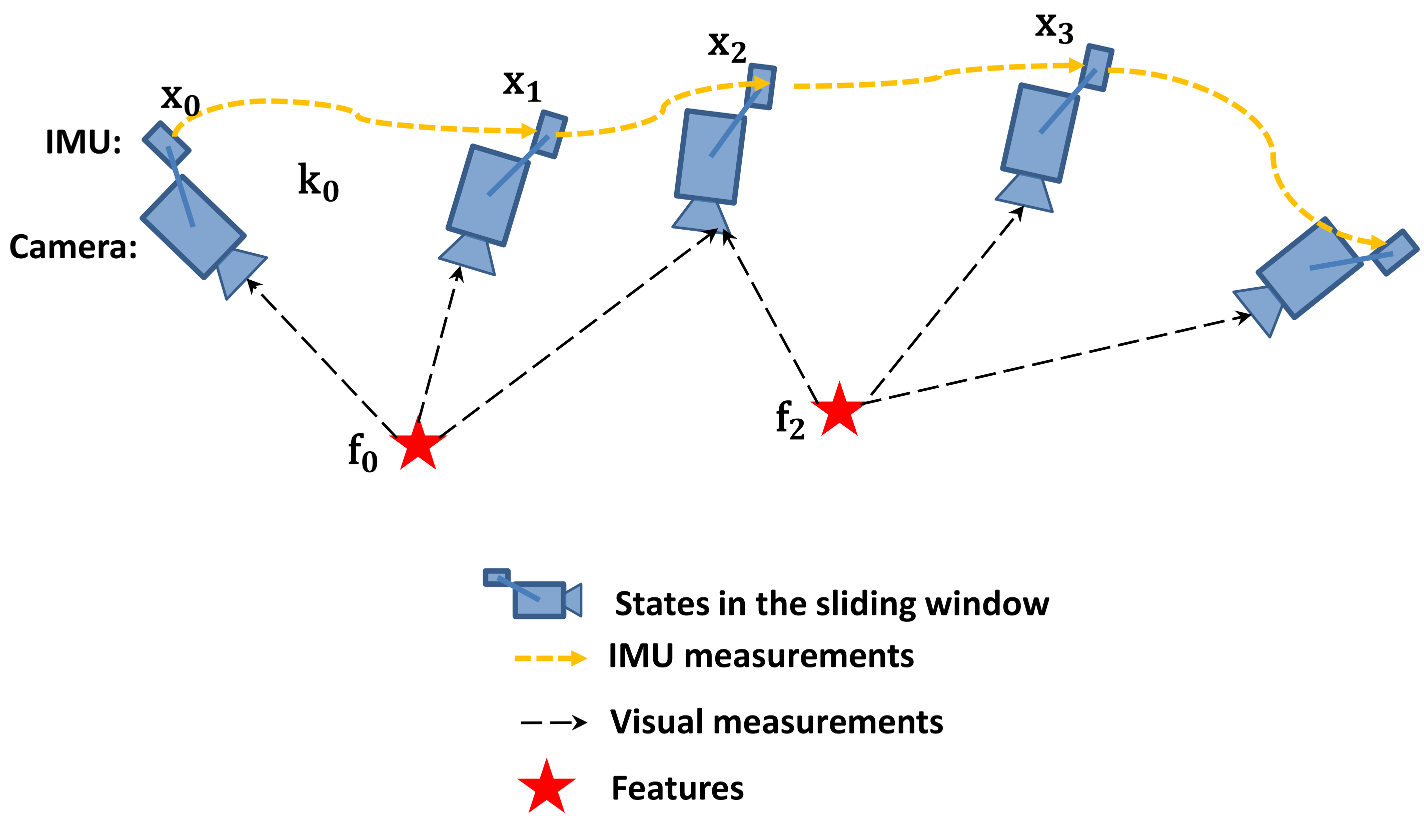
Monocular Visual-Inertial SLAM

- System diagram



Monocular Visual-Inertial Odometry

- Nonlinear graph optimization-based, tightly-coupled, sliding window, visual-inertial bundle adjustment



Monocular Visual-Inertial Odometry

- Nonlinear graph-based optimization
  - Optimize position, velocity, rotation, IMU biases, inverse feature depth, and camera-IMU extrinsic calibration simultaneously:

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

  - Minimize residuals from all sensors

IMU measurement residual   Vision measurement residual

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

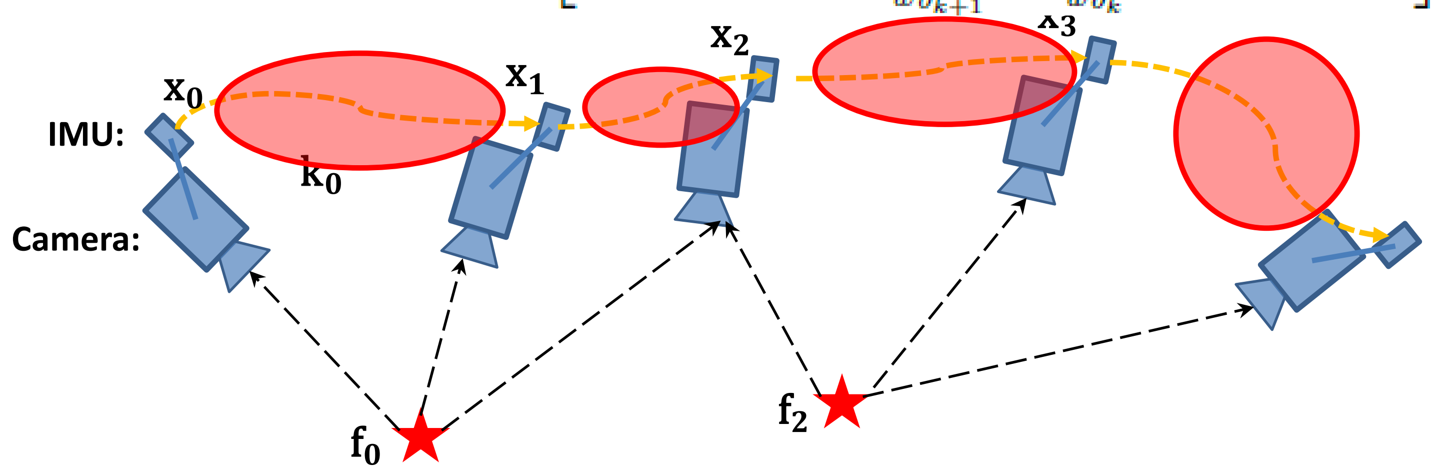
Prior from marginalization   Covariance from IMU pre-integration   Pixel reprojection covariance

Monocular Visual-Inertial Odometry

- IMU measurement residual
  - Additive for “position” and “velocity” changes, and biases
  - Multiplicative for incremental rotation

IMU pre-integration “blocks”

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



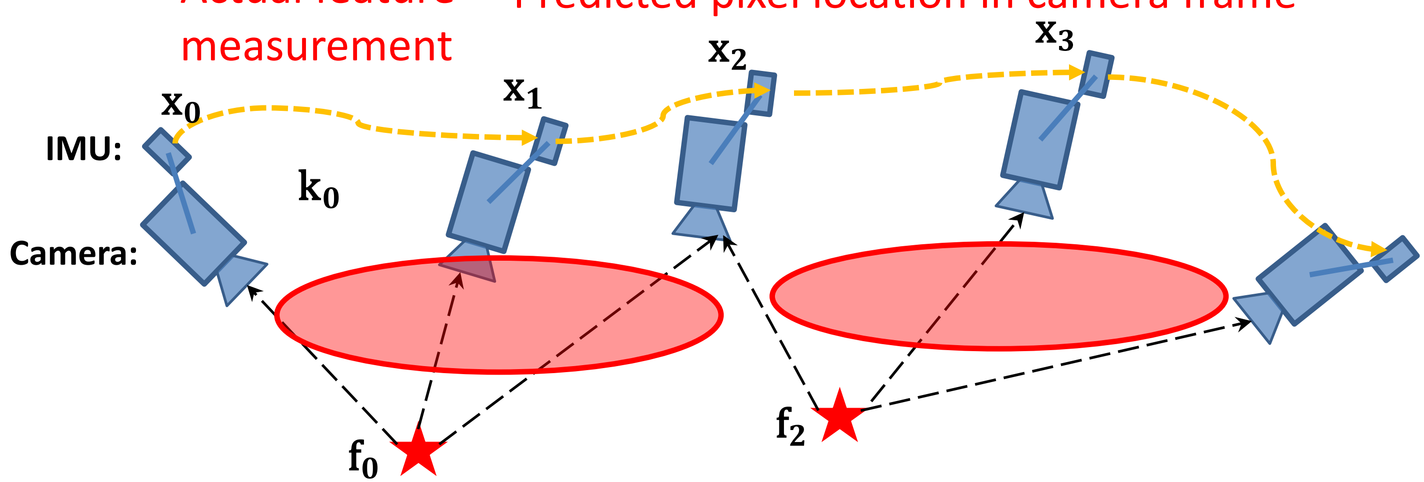
Monocular Visual-Inertial Odometry

- Vision measurement residual
  - Pixel reprojection error
  - Inverse depth model, at least 2 observations per feature, first observation to define feature direction

$$\mathbf{r}_{\mathcal{C}}(\hat{\mathbf{z}}_l^{c_j}, \mathcal{X}) = \begin{bmatrix} \hat{u}_l^{c_j} \\ \hat{v}_l^{c_j} \end{bmatrix} - \pi_c\left(\mathbf{R}_b^c\left(\mathbf{R}_w^{b_j}\left(\mathbf{R}_{b_i}^w\left(\mathbf{R}_c^b \frac{1}{\lambda_l}\pi_c^{-1}\left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix}\right) + \mathbf{p}_c^b\right) + \mathbf{p}_{b_i}^w - \mathbf{p}_{b_j}^w\right) - \mathbf{p}_c^b\right)\right)$$

Actual feature measurement

Predicted pixel location in camera frame

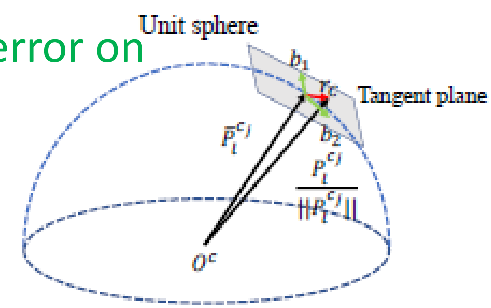


Monocular Visual-Inertial Odometry

- Vision measurement residual
  - Spherical camera model
  - At least 2 observations per feature

Reprojection error on tangent plane

$$\mathbf{r}_{\mathcal{C}}(\hat{\mathbf{z}}_l^{c_j}, \mathcal{X}) = [\mathbf{b}_1 \;\; \mathbf{b}_2]^T \cdot \left(\bar{\mathcal{P}}_l^{c_j} - \frac{\mathcal{P}_l^{c_j}}{\|\mathcal{P}_l^{c_j}\|}\right)$$

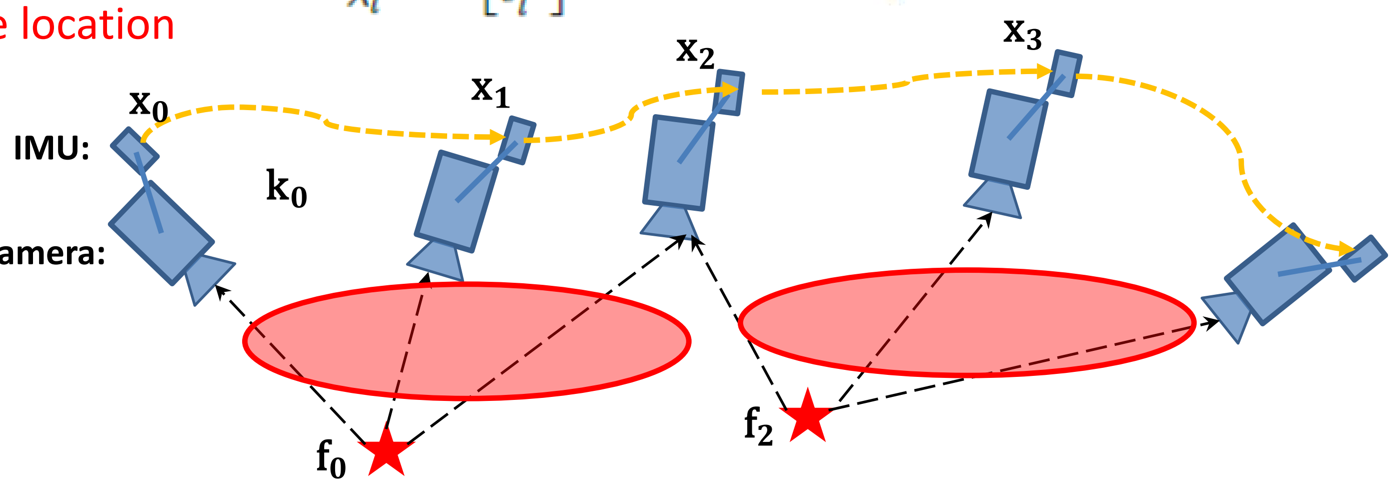


$$\bar{\mathcal{P}}_l^{c_j} = \pi_c^{-1}\left(\begin{bmatrix}\hat{u}_l^{c_j} \\ \hat{v}_l^{c_j}\end{bmatrix}\right)$$

Unit vector of feature from 2D measurement

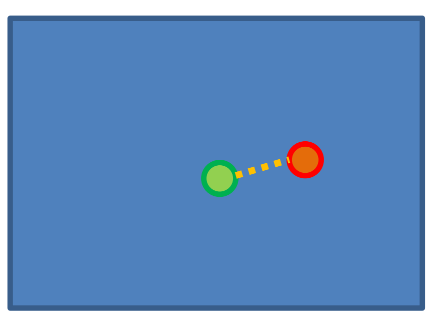
$$\mathcal{P}_l^{c_j} = \mathbf{R}_b^c(\mathbf{R}_w^{b_j}(\mathbf{R}_{b_i}^w(\mathbf{R}_c^b \frac{1}{\lambda_l}\pi_c^{-1}\left(\begin{bmatrix}u_l^{c_i} \\ v_l^{c_i}\end{bmatrix}\right) + \mathbf{p}_c^b) + \mathbf{p}_{b_i}^w - \mathbf{p}_{b_j}^w) - \mathbf{p}_c^b)$$

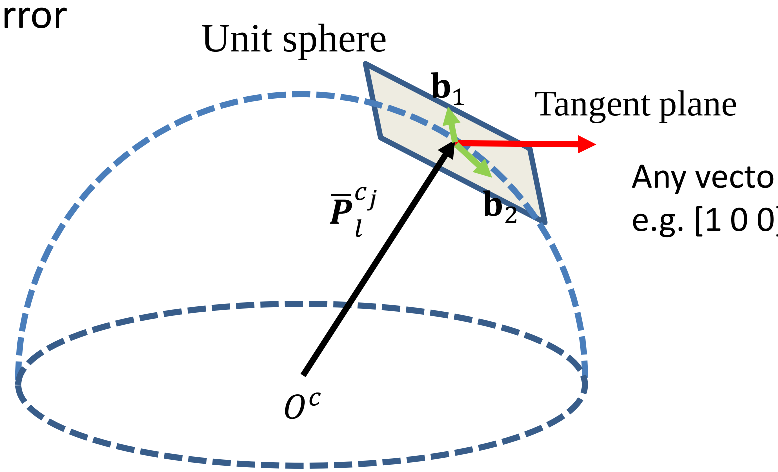
Predicted feature location in camera frame



Monocular Visual-Inertial Odometry

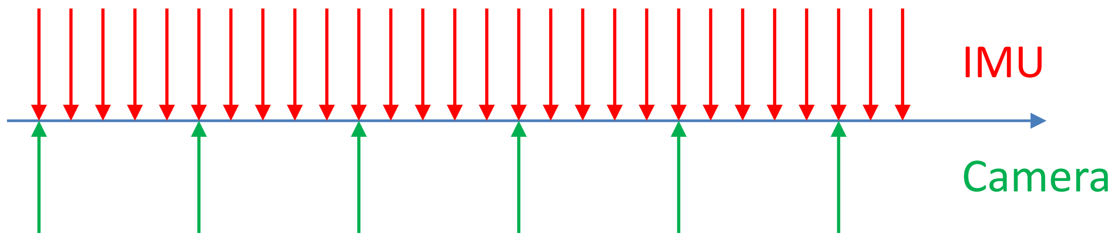
- Vision measurement residual
  - Spherical camera model
  - Finding two basis vectors on the tangent plane
    - Choose any vector not parallel with $\bar{\boldsymbol{P}}_l^{c_j}$, e.g. [1 0 0]
    - $\mathbf{b}_1 = normalize(\bar{P}_l^{c_j} \times [1\ 0\ 0])$
    - $\mathbf{b}_2 = normalize(\bar{P}_l^{c_j} \times \mathbf{b}_1)$
- Spherical vs. pinhole camera models
  - Different ways to define the reprojection error
  - Able to model cameras with arbitrary FOV



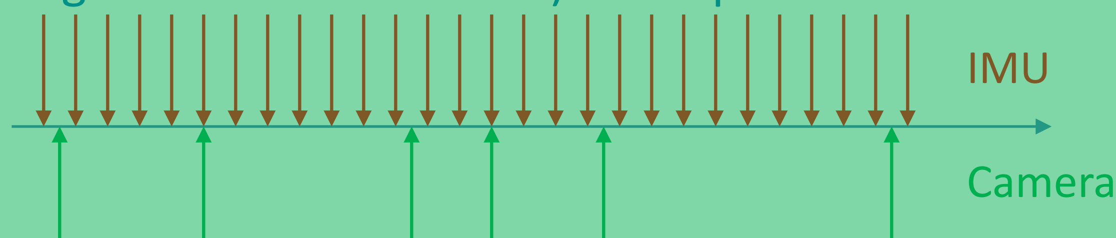


Review: Synchronization

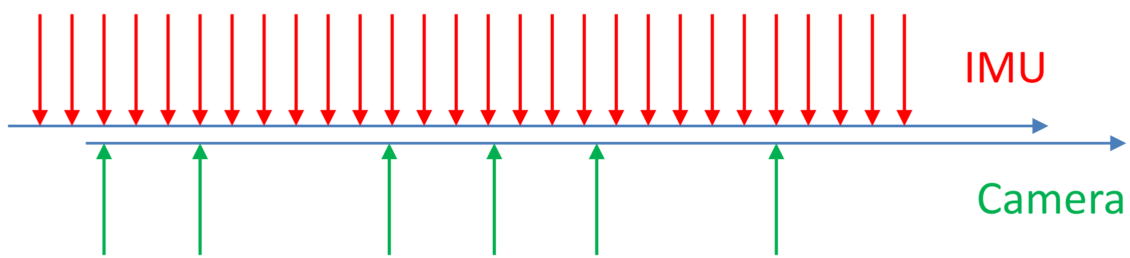
- Best: Sensors are hardware-triggered



- OK: Sensors have the same clock (e.g. running on the same system clock or have global clock correction) but capture data at different times

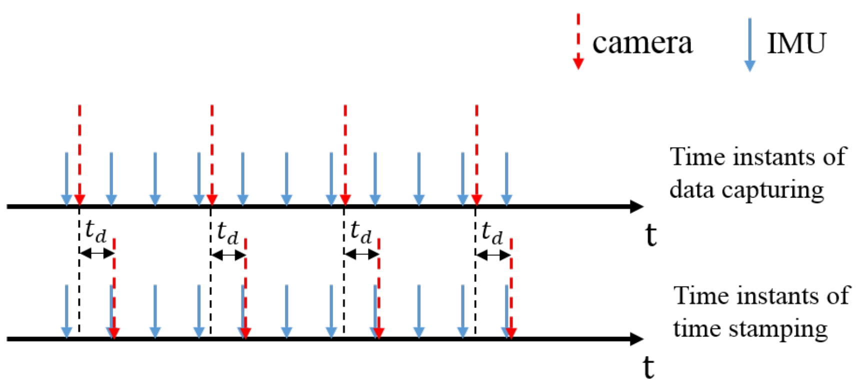


- Bad: Sensors have different clocks (e.g. each sensor has its own oscillator)

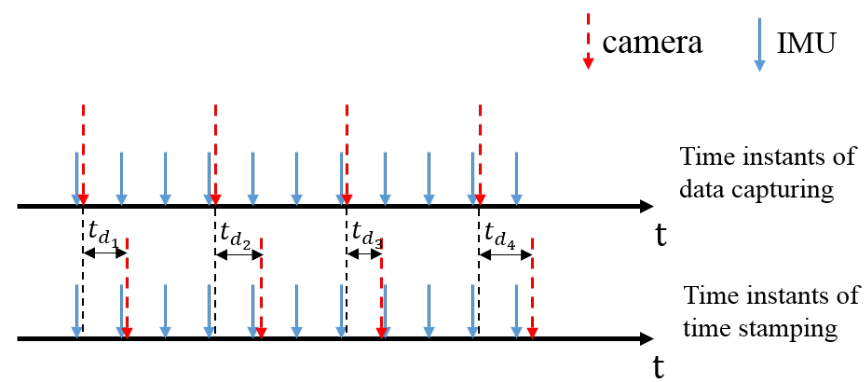


Review: Timestamps

- Timestamp: how the time for each sensor measurement is tagged
- Best: timestamping is done at data capture
- OK: fixed latency for time stamping
  - e.g. time is tagged on low-level hardware after some fixed-duration data processing, and will not be affected by any dynamic OS scheduling tasks
- Bad: variable latency in time stamping
  - e.g. plug two sensors into USB ports and time stamp according to the PC time. Time stamping is affected by data transmission latency from the sensor to PC



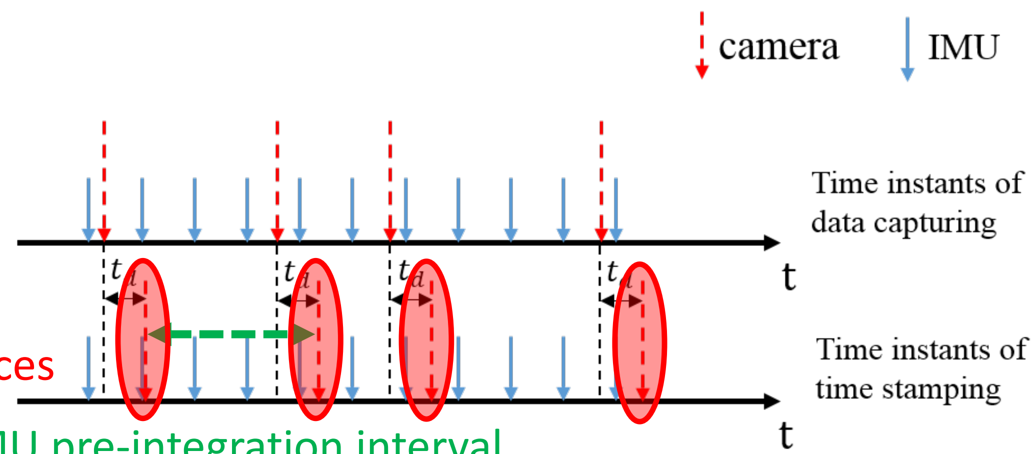
Good



Bad

Monocular Visual-Inertial Odometry

- Temporal calibration
  - Calibrate the fixed latency t_d occurred during time stamping
  - Change the IMU pre-integration interval to the interval between two image timestamps
    - Linear incorporation of IMU measurements to obtain the IMU reading at image time stamping
    - Estimates states (position, orientation, etc.) at image time stamping



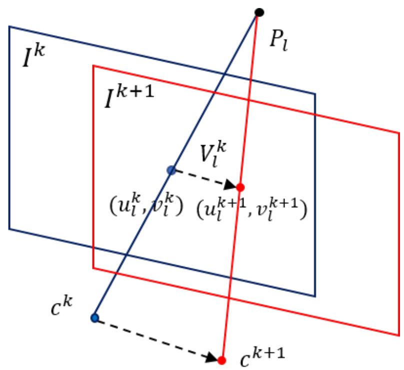
Estimate states at these time instances

IMU pre-integration interval

Monocular Visual-Inertial Odometry

- Vision measurement residual for temporal calibration
  - Feature velocity on image plane
    - feature l moves at speed V_l^k from image k to $k+1$ in short time period $[t_k, t_{k+1}]$

$$\mathbf{V}_l^k = \left(\begin{bmatrix} u_l^{k+1} \\ v_l^{k+1} \end{bmatrix} - \begin{bmatrix} u_l^k \\ v_l^k \end{bmatrix}\right) / (t_{k+1} - t_k)$$

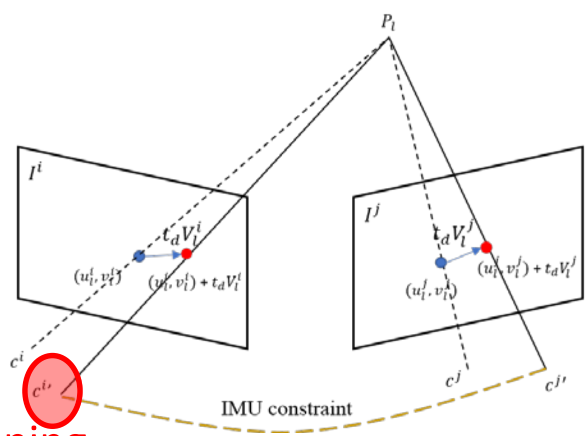


  - Visual measurement residual with time offset
    - New state variable t_d, and estimate states $(c^{i'}, c^{j'})$ at time stamping

$$\mathbf{r}_{\mathcal{C}}(\hat{\mathbf{z}}_l^{c_j}, \mathcal{X}) = [\mathbf{b}_1 \;\; \mathbf{b}_2]^T \cdot (\bar{\mathcal{P}}_l^{c_j} - \frac{\mathcal{P}_l^{c_j}}{\|\mathcal{P}_l^{c_j}\|})$$

$$\bar{\mathcal{P}}_l^{c_j} = \pi_c^{-1}\left(\begin{bmatrix} \hat{u}_l^{c_j} \\ \hat{v}_l^{c_j} \end{bmatrix} + t_d \vec{V}_l^{c_j}\right)$$

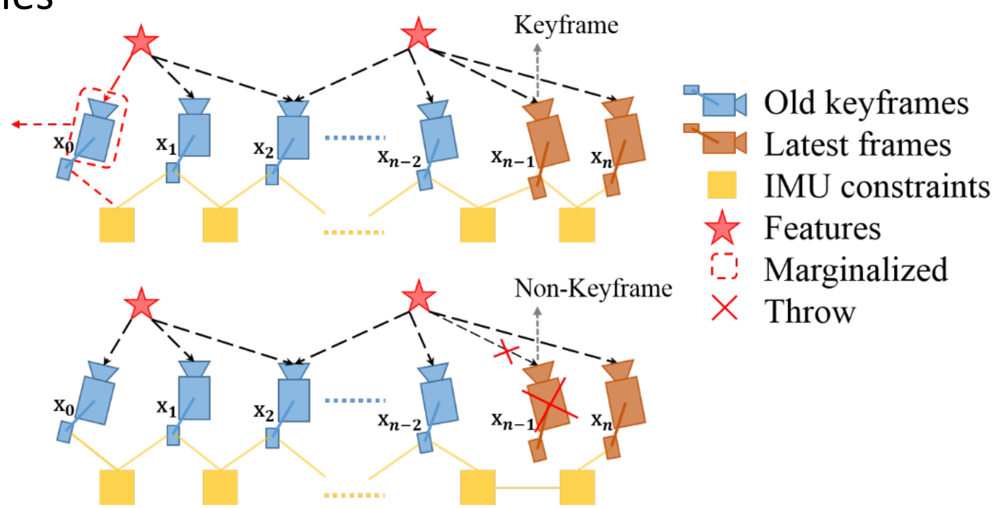
$$\mathcal{P}_l^{c_j} = \mathbf{R}_b^c(\mathbf{R}_w^{b_j}(\mathbf{R}_{b_i}^w(\mathbf{R}_c^b \frac{1}{\lambda_l} \pi_c^{-1}\left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \end{bmatrix} + t_d \vec{V}_l^{c_i}\right) + \mathbf{p}_c^b) + \mathbf{p}_{b_i}^w - \mathbf{p}_{b_j}^w) - \mathbf{p}_c^b)$$



"Virtual image" at time stamping

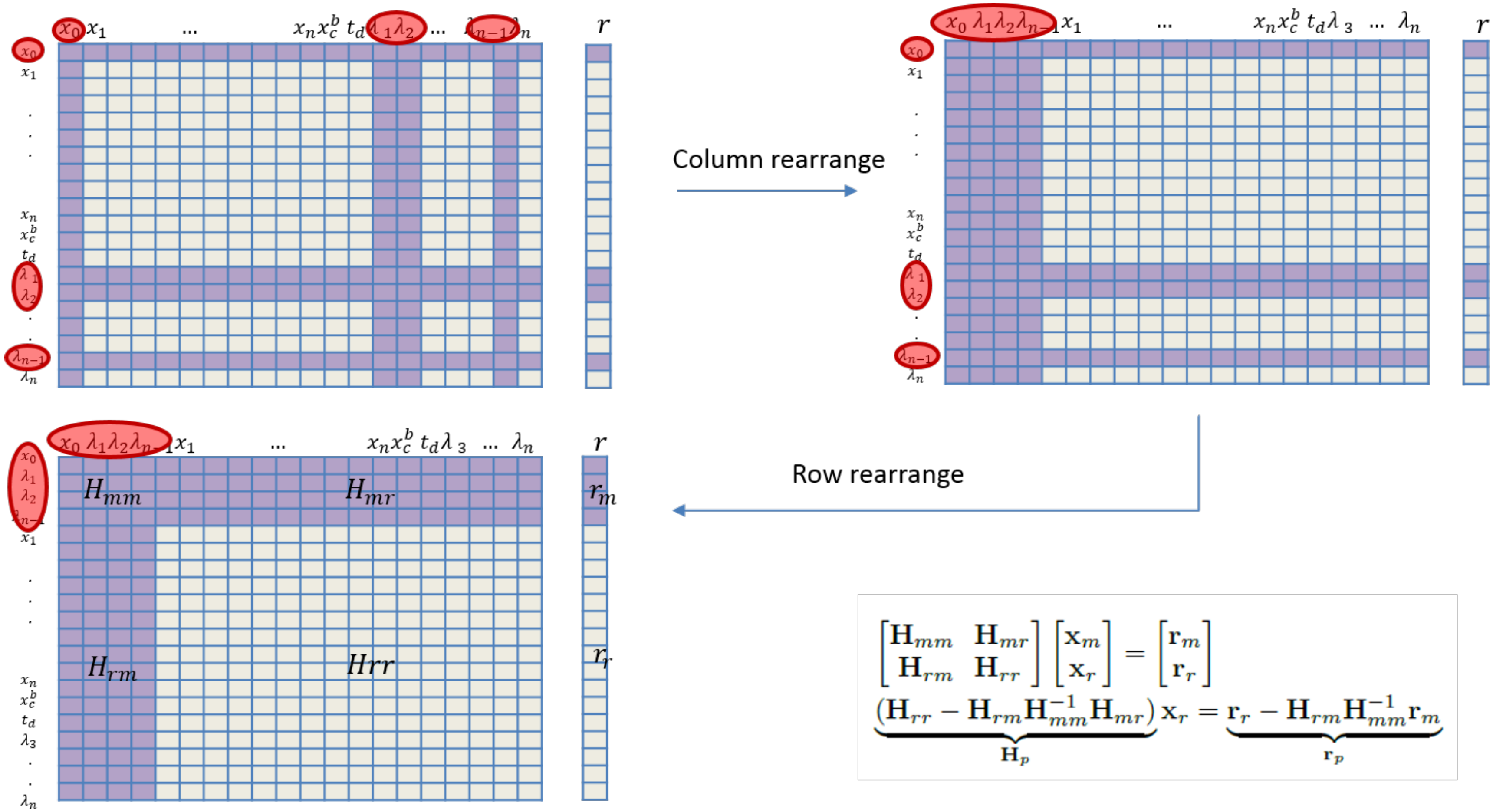
Monocular Visual-Inertial Odometry

- Marginalization
  - Bound computation complexity to a sliding window of states
  - Basic principles:
    - Add all frames into the sliding window, and remove non-keyframes after the nonlinear optimization
    - keep as many keyframes with sufficient parallax as possible
    - Maintain matrix sparsity by throwing away visual measurements from non-keyframes



Monocular Visual-Inertial Odometry

- Marginalization via Schur complement on information matrix



$$\begin{bmatrix} \mathbf{H}_{mm} & \mathbf{H}_{mr} \\ \mathbf{H}_{rm} & \mathbf{H}_{rr} \end{bmatrix} \begin{bmatrix} \mathbf{x}_m \\ \mathbf{x}_r \end{bmatrix} = \begin{bmatrix} \mathbf{r}_m \\ \mathbf{r}_r \end{bmatrix}$$

$$\underbrace{(\mathbf{H}_{rr} - \mathbf{H}_{rm}\mathbf{H}_{mm}^{-1}\mathbf{H}_{mr})}_{\mathbf{H}_p} \mathbf{x}_r = \underbrace{\mathbf{r}_r - \mathbf{H}_{rm}\mathbf{H}_{mm}^{-1}\mathbf{r}_m}_{\mathbf{r}_p}$$

Monocular Visual-Inertial Odometry

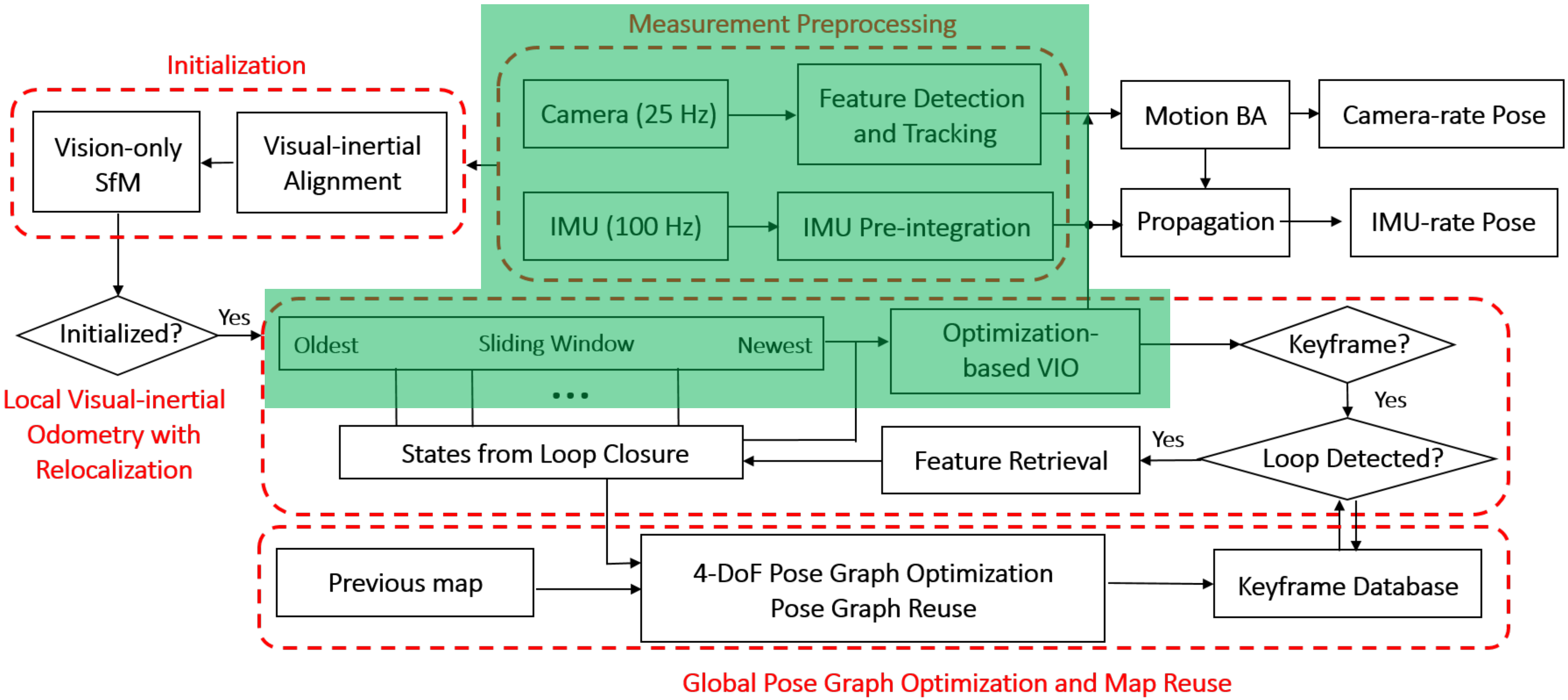
- Solving the nonlinear system
  - Minimize residuals from all sensors

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

  - Linearize (to Ax=b), solve, and iterate until time budget is reached
  - Ceres Solver (http://ceres-solver.org/)
  - Utilize sparse matrix solver
- Qualitative discussion on solution quality
  - Numerical stability issues always exist, much worse than vSLAM
    - Good: walking and aerial robots
    - Bad: ground vehicle moving in 2D
    - Failure: constant velocity or pure rotation
  - Downgraded performance in distanced scenes

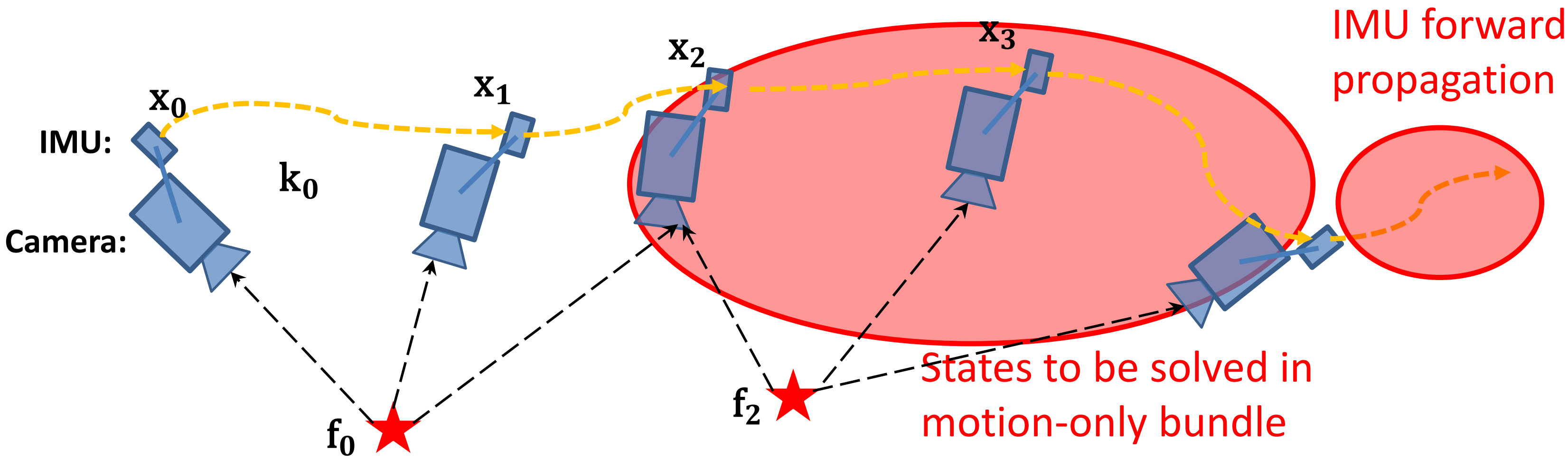
Monocular Visual-Inertial SLAM

- System diagram



Monocular Visual-Inertial Odometry

- Speeding up
  - The sliding window monocular visual-inertial bundle adjustment runs at 10Hz
  - Motion-only visual-inertial bundle adjustment to boost up the state estimation 30Hz
  - IMU forward propagation to boost to 100Hz



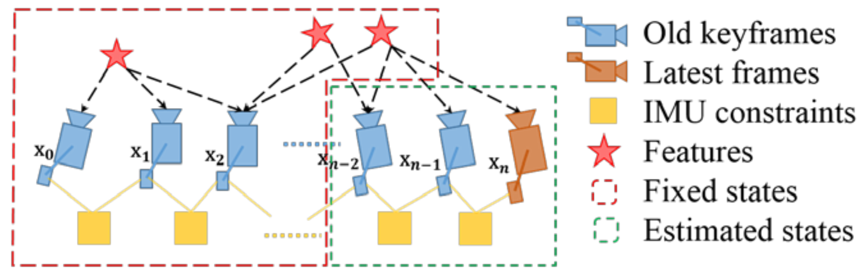
Monocular Visual-Inertial Odometry

- Motion-only visual-inertial bundle adjustment
  - Optimize position, velocity, rotation in a smaller windows, assuming all other quantities are fixed

$$\mathcal{X} = [\cancel{\mathbf{x}_0, \mathbf{x}_1,} \cdots \mathbf{x}_n, \cancel{\mathbf{x}_c^b, \lambda_0, \lambda_1, \cdots \lambda_m}]$$

$$\mathbf{x}_k = [\mathbf{p}^w_{b_k}, \mathbf{v}^w_{b_k}, \mathbf{q}^w_{b_k}, \cancel{\mathbf{b}_a, \mathbf{b}_g}], k \in [0, n]$$

$$\cancel{\mathbf{x}_c^b = [\mathbf{p}_c^b, \mathbf{q}_c^b]},$$



  - Prior in cost function is ignored

IMU measurement residual Vision measurement residual

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

Covariance from IMU pre-integration Pixel reprojection covariance

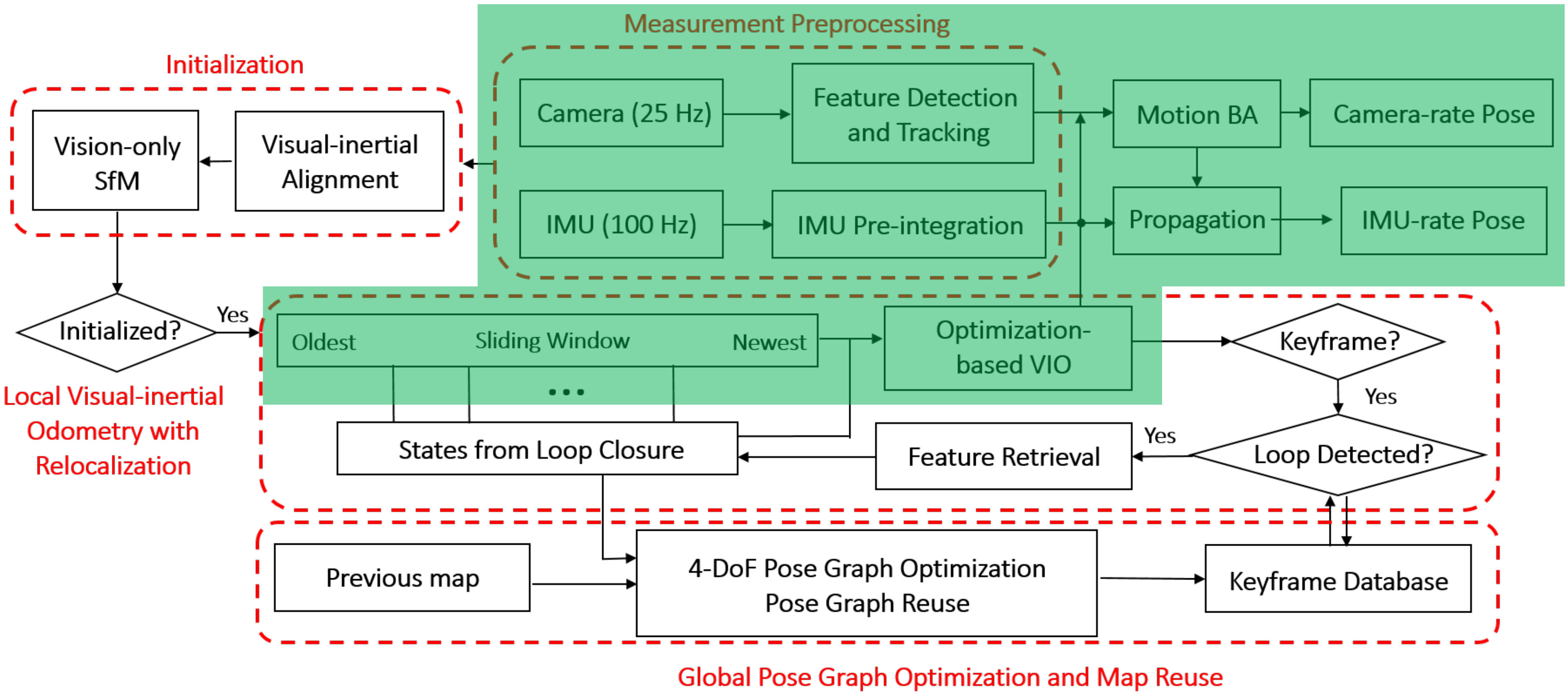
  - Also solved using the Ceres Solver

Monocular Visual-Inertial Odometry

- Failure detection
  - Few trackable feature in the current frame
  - Large jumps in nonlinear solver
  - Abnormal bias or extrinsic parameter calibration
  - Modeled as a standalone module, more to be added…
- Failure recovery
  - Just run the initialization again…
  - Lots of book keeping…

Monocular Visual-Inertial SLAM

- System diagram

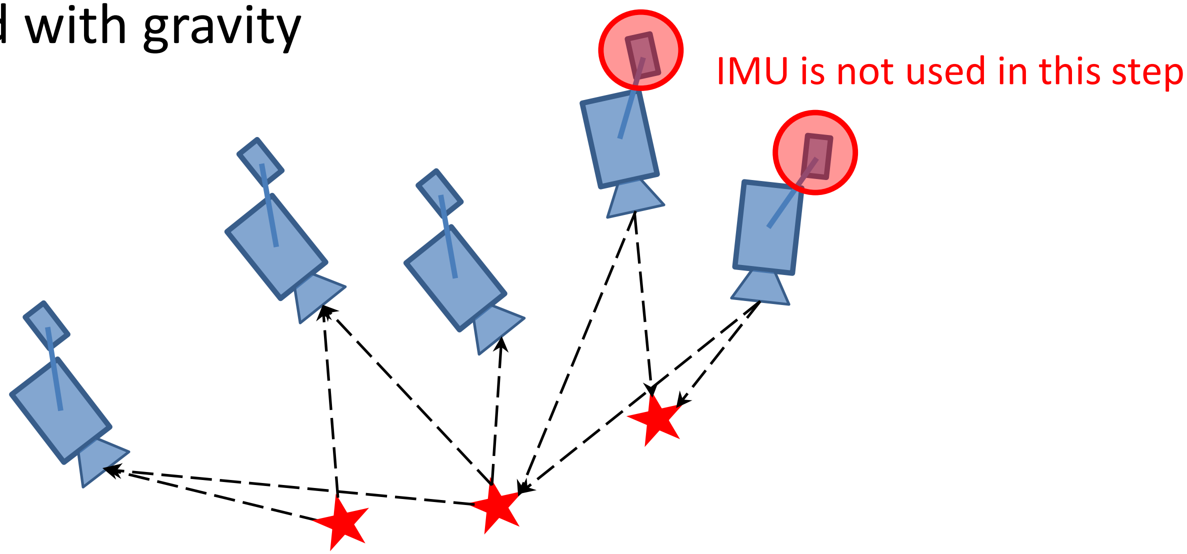


Estimator Initialization

- Very, very, very important for monocular visual-inertial systems
- Assumption 1: known camera-IMU extrinsic calibration during initialization
  - Does not need to be very accurate
  - Extrinsic calibration is refined in later nonlinear optimization
- Assumption 2: known accelerometer and gyroscope biases during initialization
  - Use zero values at power-up
  - Use prior values during failure recovery
  - Reasonable assumption due to slow varying nature of biases
- Pipeline
  - Monocular vision-only SFM in a local window
  - Visual-inertial alignment

Estimator Initialization

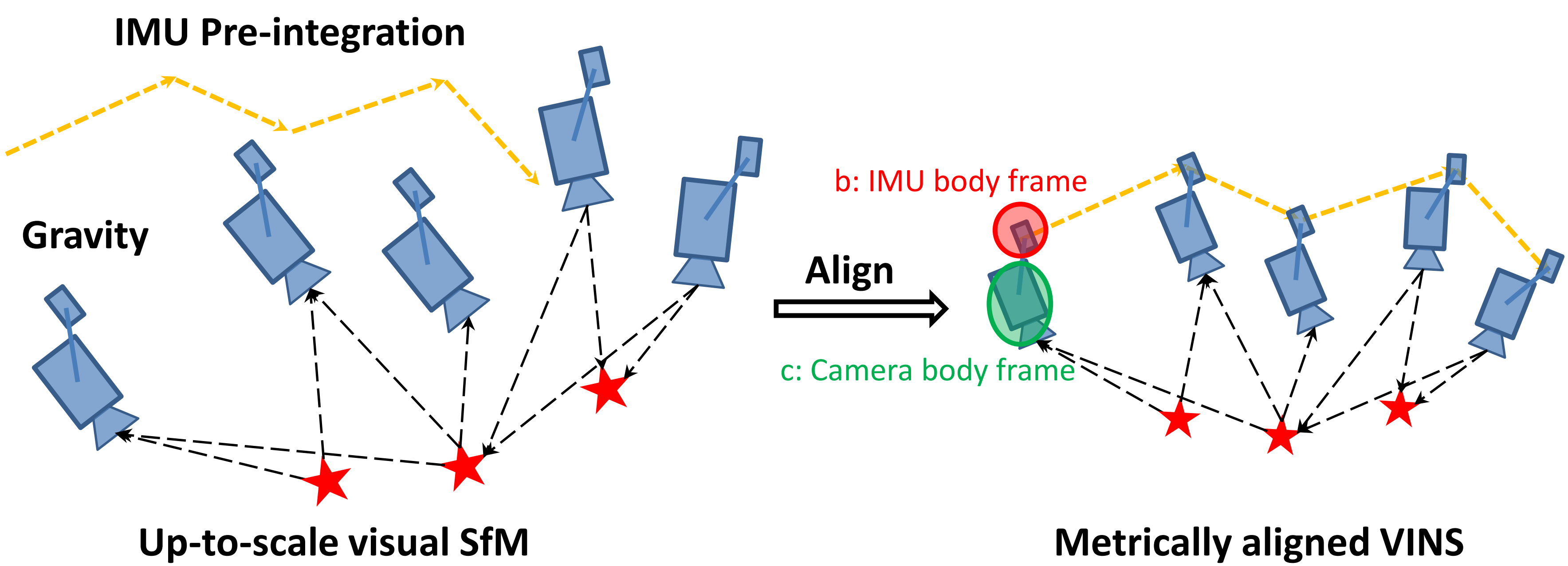
- Monocular vision-only structure-from-motion (SfM)
  - In a small window (10 frames, 1sec)
  - Up-to-scale, locally drift-free position estimates
  - Locally drift-free orientation estimates
  - Not aligned with gravity



Estimator Initialization

- Visual-inertial alignment
  - Estimates velocity of each frame, gravity vector, and scale
    - Note the coordinate frames

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



Estimator Initialization

- Visual-inertial alignment
  - Linear measurement model

IMU Pre-integration

Known values from vSfM and extrinsic calibration

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

$$\approx \begin{bmatrix} -\mathbf{R}^{b_k}_{c_0}\Delta t_k & \mathbf{0} & \frac{1}{2}\mathbf{R}^{b_k}_{c_0}\Delta t_k^2 & \mathbf{R}^{b_k}_{c_0}(\bar{\mathbf{p}}^{c_0}_{c_{k+1}} - \bar{\mathbf{p}}^{c_0}_{c_k}) \\ -\mathbf{R}^{b_k}_{c_0} & \mathbf{R}^{b_k}_{c_0} & \mathbf{R}^{b_k}_{c_0}\Delta t_k & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}^{c_0}_{b_k} \\ \mathbf{v}^{c_0}_{b_{k+1}} \\ \mathbf{g}^{c_0} \\ s \end{bmatrix}$$

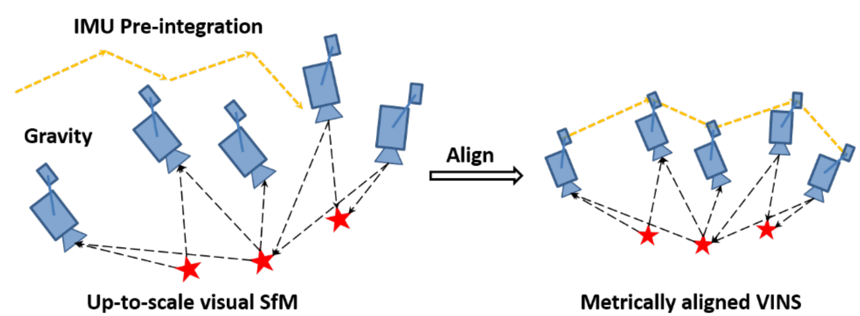
States to be initialized

Up-to-scale translation from vSfM

  - Solve a linear system
    - Scale and rotate the vSfM

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

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

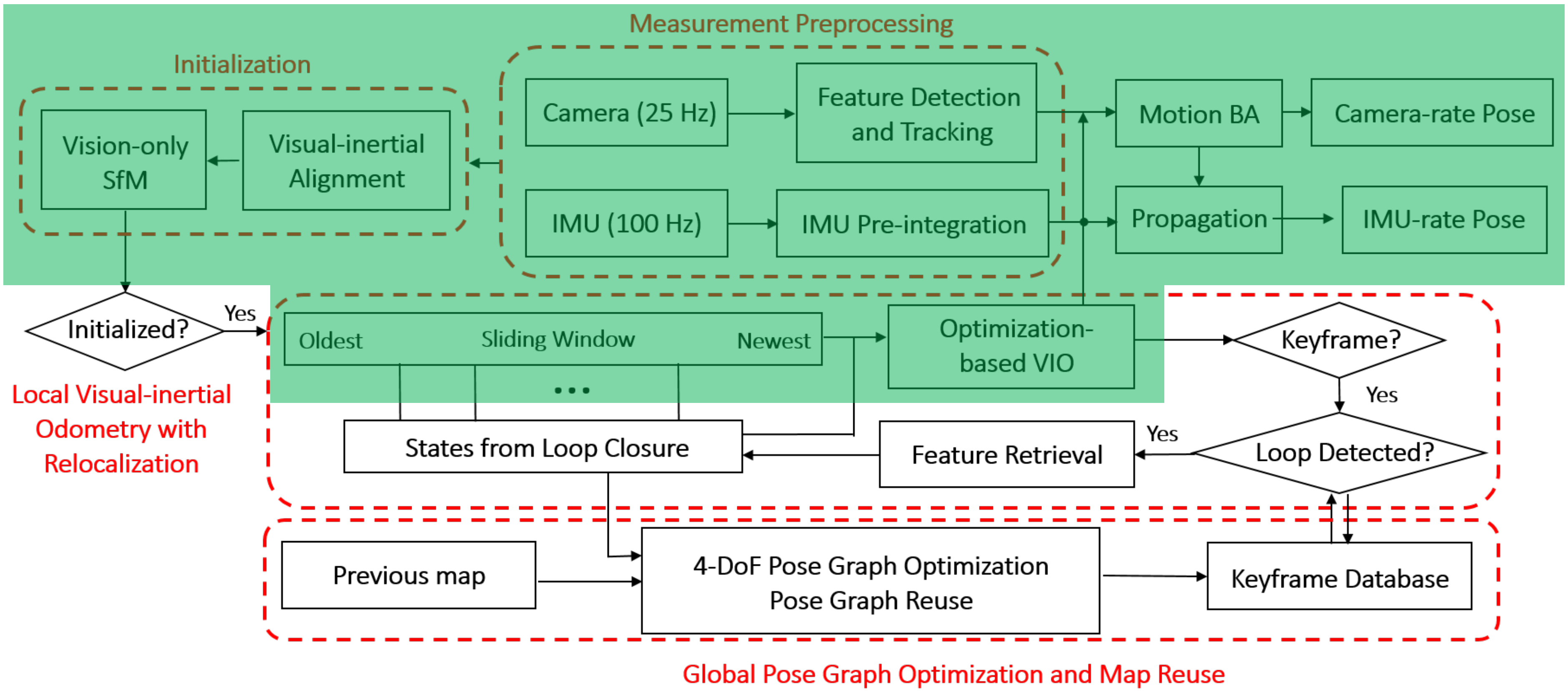


Estimator Initialization

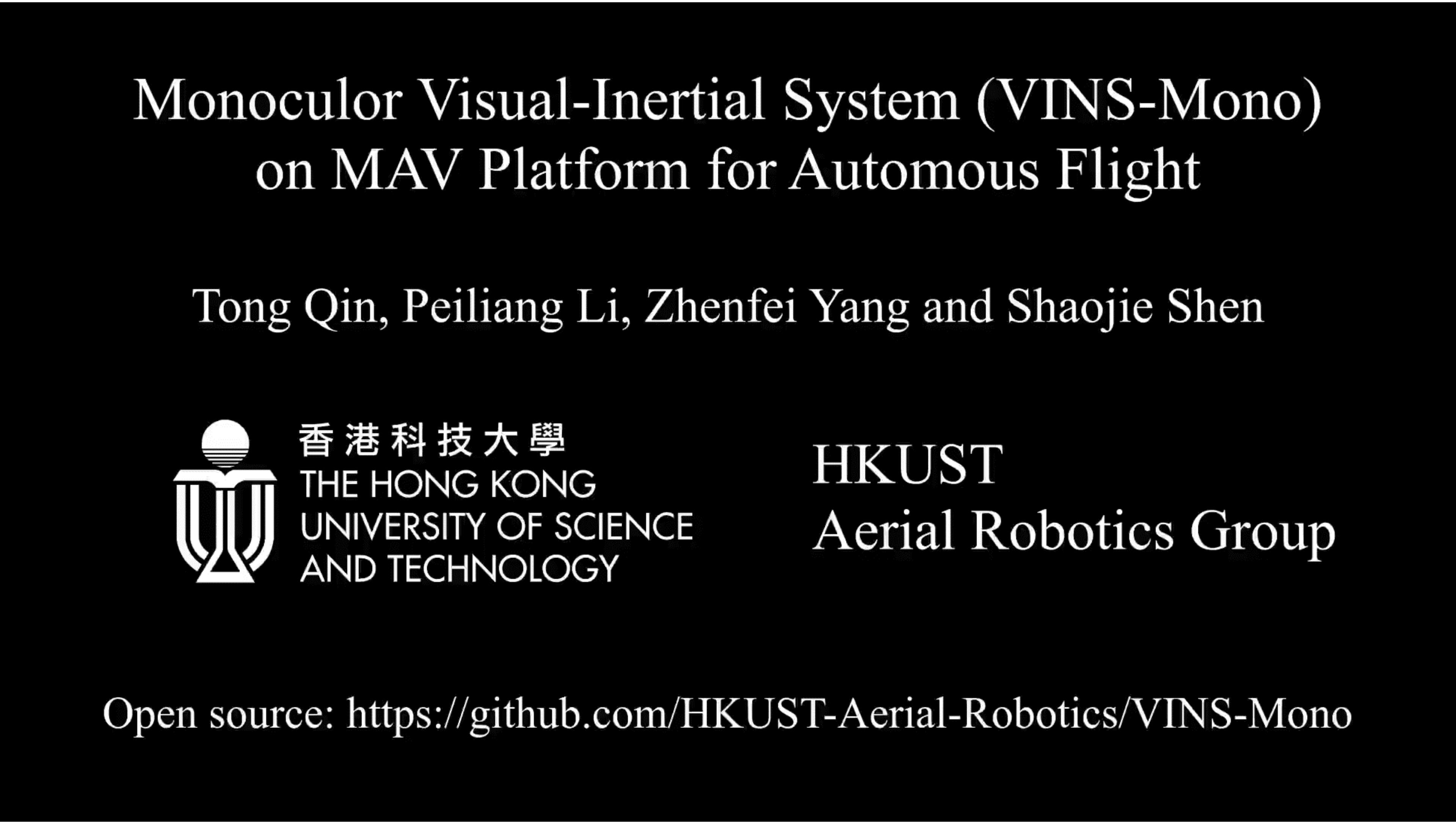
- Current issues:
  - IMU biases are not initialized
    - Gyroscope: obtained from stationary measurements
    - Accelerometer: problematic…
  - May fail at high altitude scenes due to excessive IMU integration time
    - Solution: Spline-based initialization, use derivatives instead of integration
    - T. Liu and S. Shen. High altitude monocular visual-inertial state estimation: initialization and sensor fusion. In Proc. of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, May 2017

Monocular Visual-Inertial SLAM

- System diagram

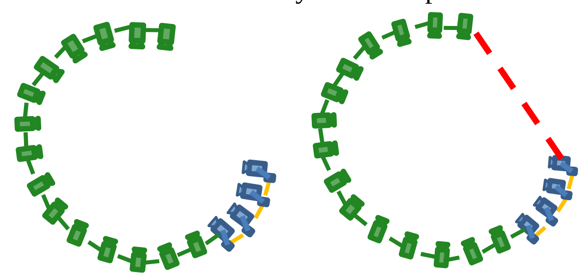


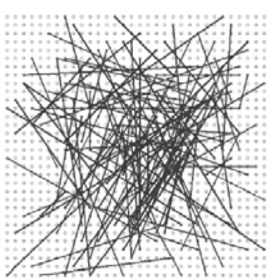
Visual-Inertial SLAM for Autonomous Drone

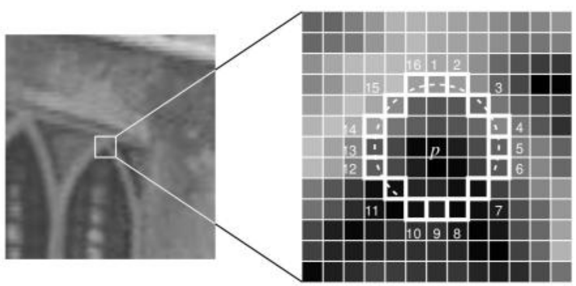


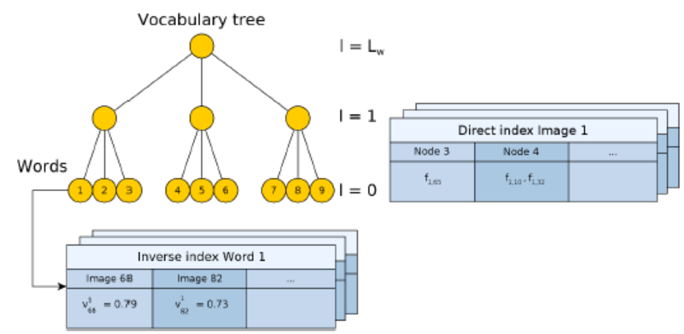
Loop Closure

- Loop detection
  - Describe features by BRIEF
    - Features that we use in the VIO (200, not enough for loop detection)
    - Extract new FAST features (500, only use for loop detection)
  - Query Bag-of-Word (DBoW2)
    - Return loop candidates





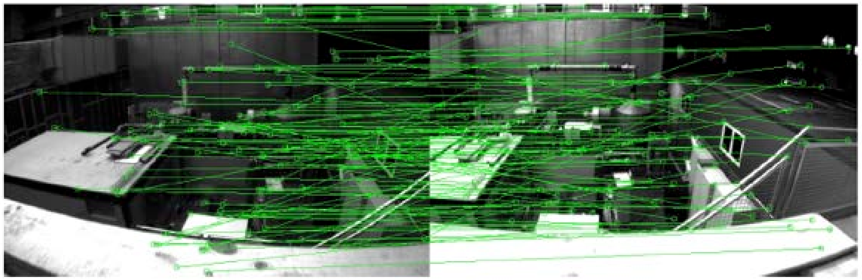




Calonder, Michael, et al. "Brief: Binary robust independent elementary features." *Computer Vision–ECCV 2010* (2010): 778-792.
Gálvez-López, Dorian, and Juan D. Tardos. "Bags of binary words for fast place recognition in image sequences." *IEEE Transactions on Robotics* 28.5 (2012): 1188-1197.

Loop Closure

- Feature Retrieving
  - Try to retrieve matches for features (200) that are used in the VIO
  - BRIEF descriptor match
  - Geometric check
    - Fundamental matrix test with RANSAC
    - At least 30 inliers
- Output:
  - Loop closure frames with known pose
  - Feature matches between VIO frames and loop closure frames

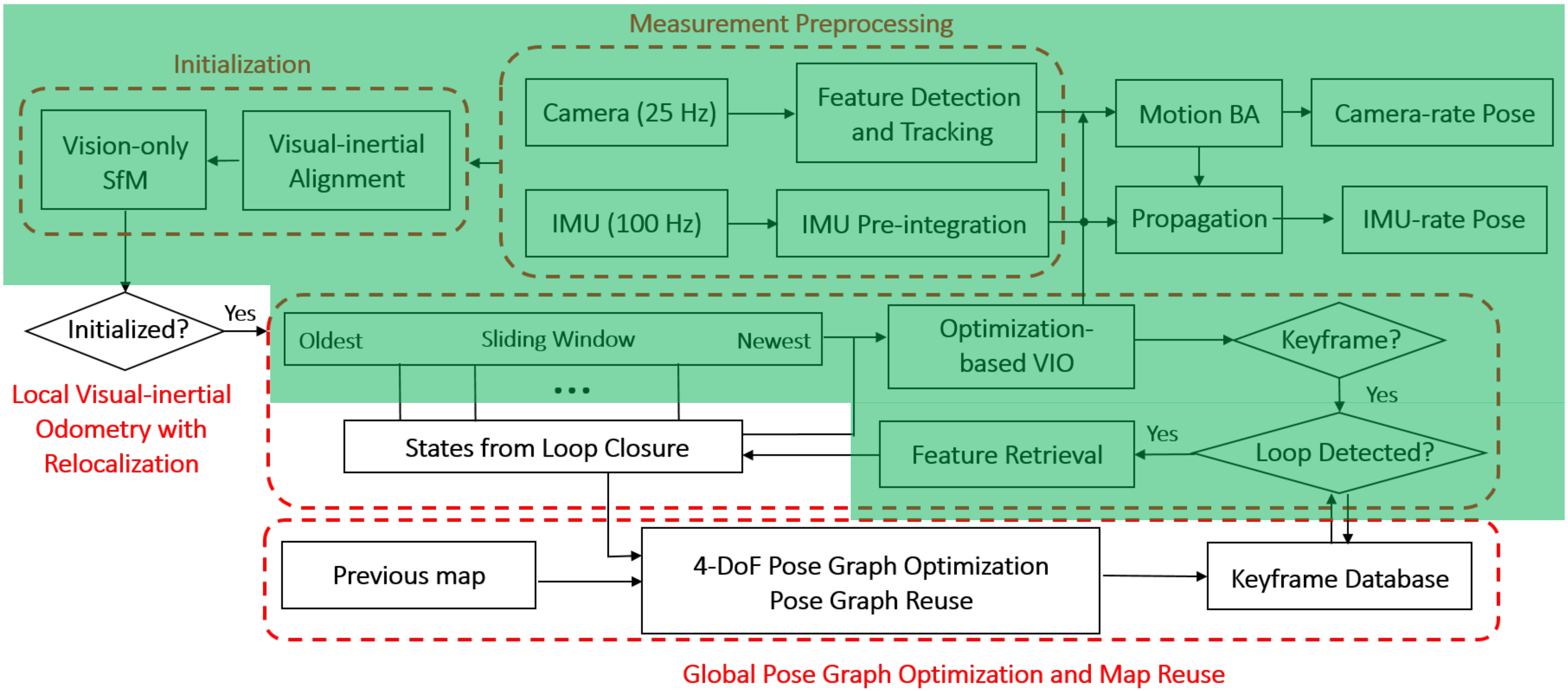


(a)

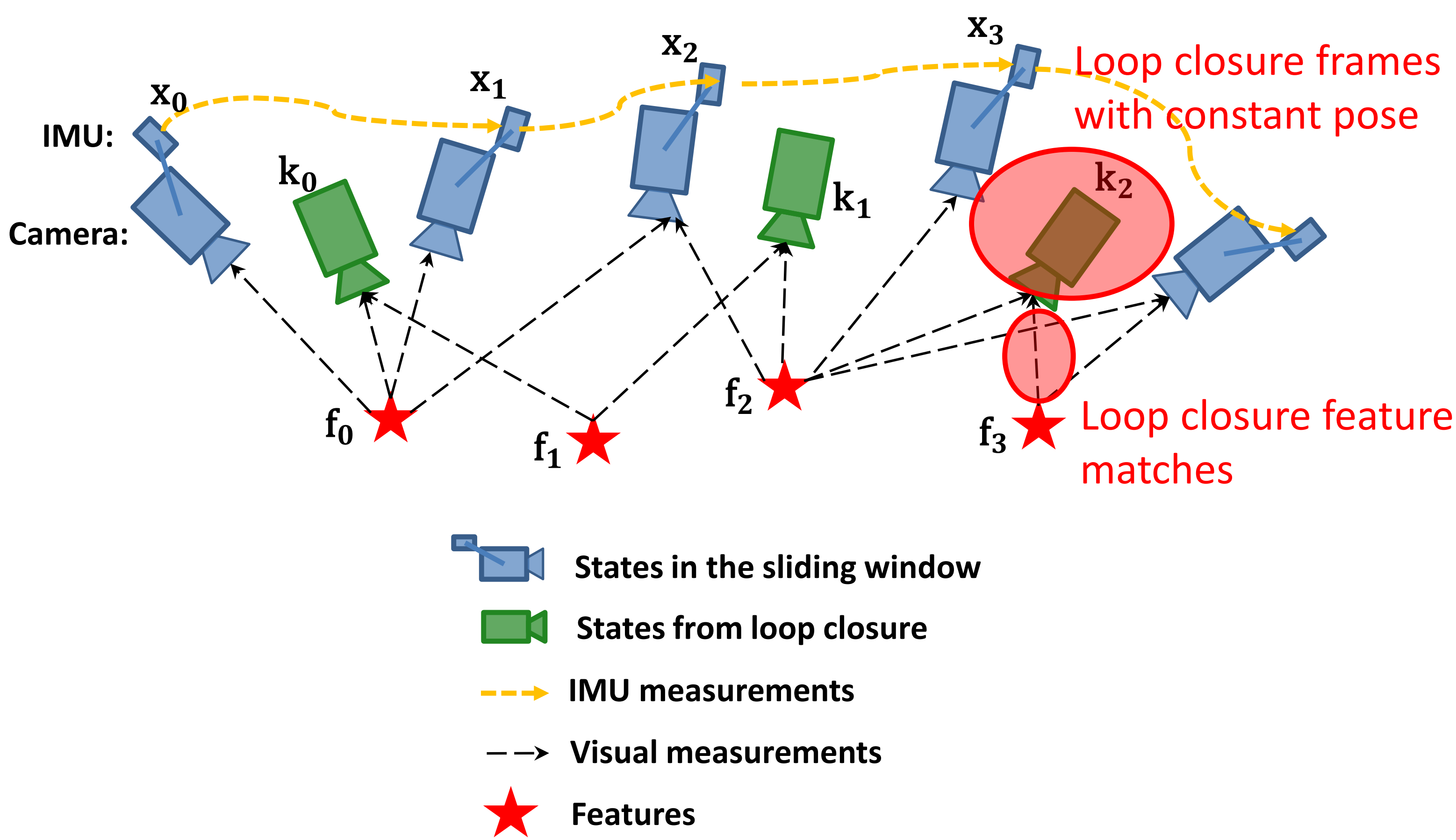
(b)

Monocular Visual-Inertial SLAM

- System diagram

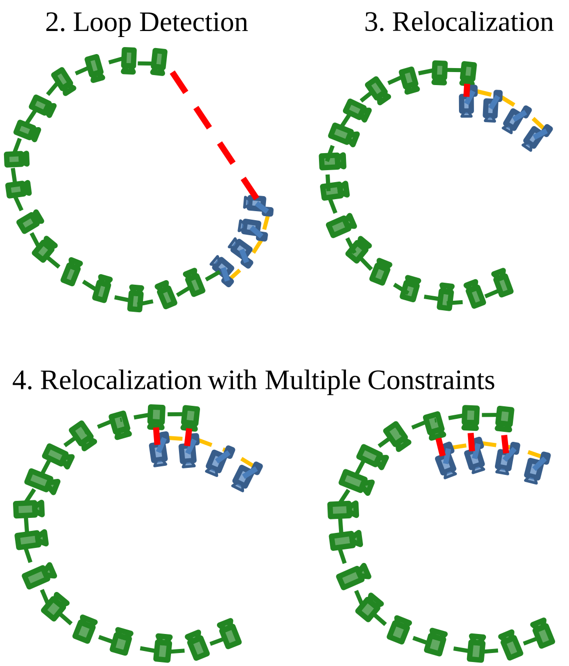


Monocular Visual-Inertial Odometry with Relozalization



Monocular Visual-Inertial Odometry with Relozalization

- Relocalization
  - Visual measurements for tightly-coupled relocalization
    - Observation of retrieved features in loop closure frames
    - Poses of loop closure frames are constant
    - No increase in state vector dimension for relocalization
    - Allows multi-constraint relocalization



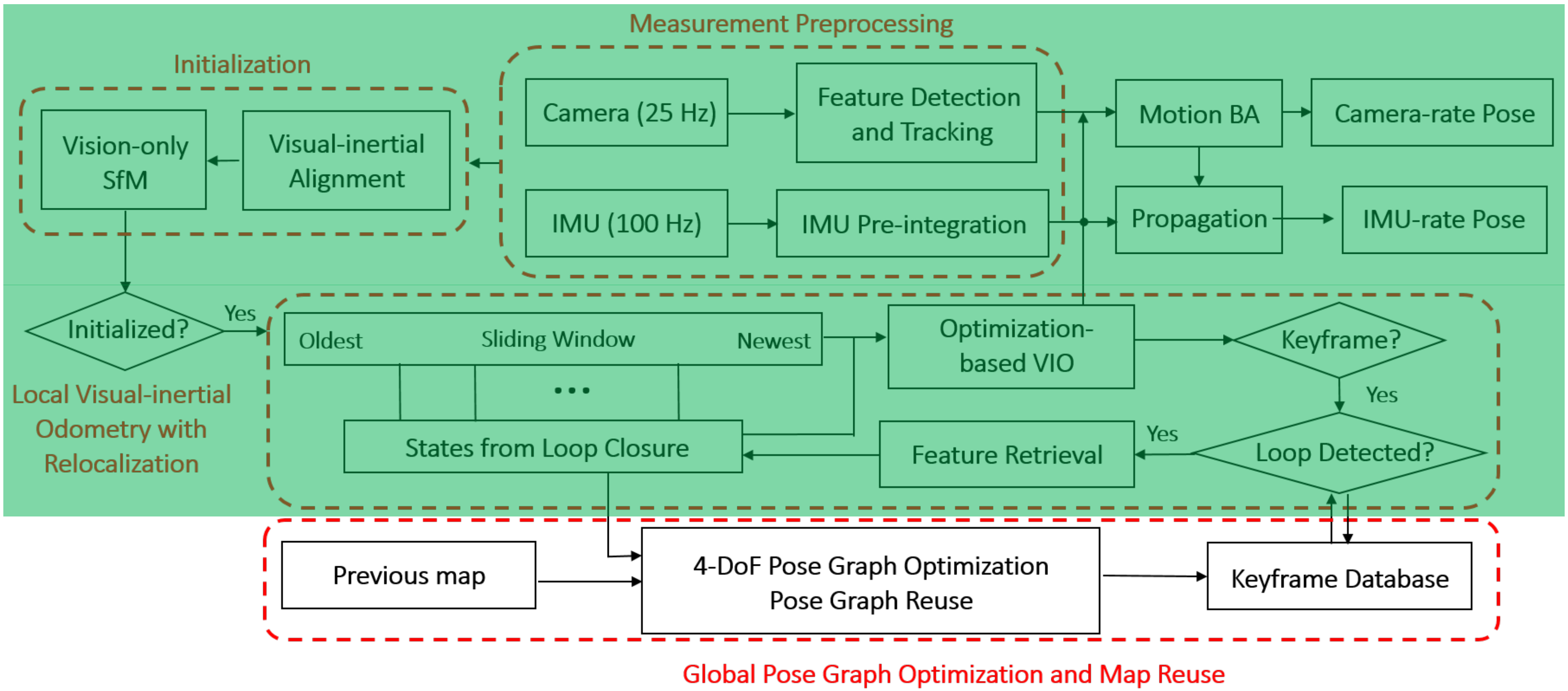
$$\min_{\mathcal{X}} \left\{ \left\|\mathbf{r}_p - \mathbf{H}_p \mathcal{X}\right\|^2 + \sum_{k\in\mathcal{B}} \left\|\mathbf{r}_\mathcal{B}(\hat{\mathbf{z}}^{b_k}_{b_{k+1}}, \mathcal{X})\right\|^2_{\mathbf{P}^{b_k}_{b_{k+1}}} + \sum_{(l,j)\in\mathcal{C}} \left\|\mathbf{r}_\mathcal{C}(\hat{\mathbf{z}}^{c_j}_l,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

VIO residuals

Loop closure vision measurement residual

Monocular Visual-Inertial SLAM

- System diagram



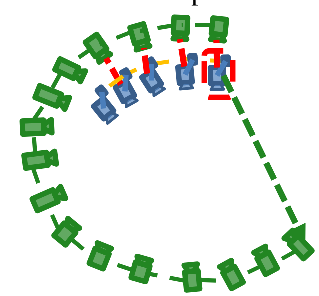
Global Pose Graph SLAM

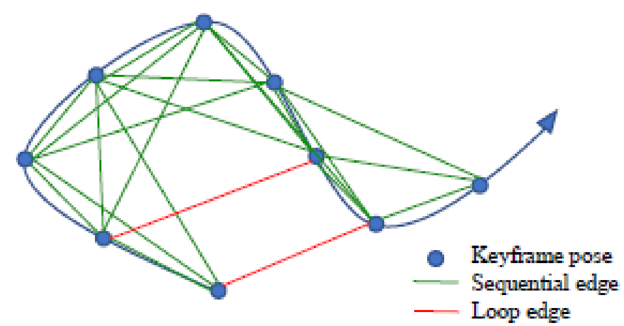
- 4-DOF pose graph
  - Roll and pitch are observable from VIO
- Adding keyframes into pose graph

$$\hat{\mathbf{p}}_{ij}^{i} = \hat{\mathbf{R}}_{i}^{w^{-1}}(\hat{\mathbf{p}}_{j}^{w} - \hat{\mathbf{p}}_{i}^{w})$$
$$\hat{\psi}_{ij} = \hat{\psi}_{j} - \hat{\psi}_{i}$$

  - Sequential edges from VIO
    - Connected with 4 previous keyframes
  - Loop closure edges
    - Only added when a keyframe is marginalized out from the sliding window VIO
    - Multi-constraint relocalization helps eliminating false loop closures

5. Add Keyframe into Pose Graph





Global Pose Graph SLAM

- 4-DOF relative pose residual:

Observable attitude from VIO

$$\mathbf{r}_{i,j}(\mathbf{p}_i^w, \psi_i, \mathbf{p}_j^w, \psi_j) = \begin{bmatrix} \mathbf{R}(\hat{\phi}_i, \hat{\theta}_i, \psi_i)^{-1}(\mathbf{p}_j^w - \mathbf{p}_i^w) - \hat{\mathbf{p}}_{ij}^i \\ \psi_j - \psi_i - \hat{\psi}_{ij} \end{bmatrix}$$

- Minimize the following cost function
  - Sequential edge from VIO
  - Loop closure edges
    - Huber norm for rejection of wrong loops

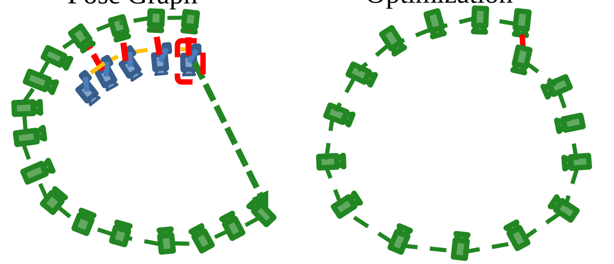
Sequential edges

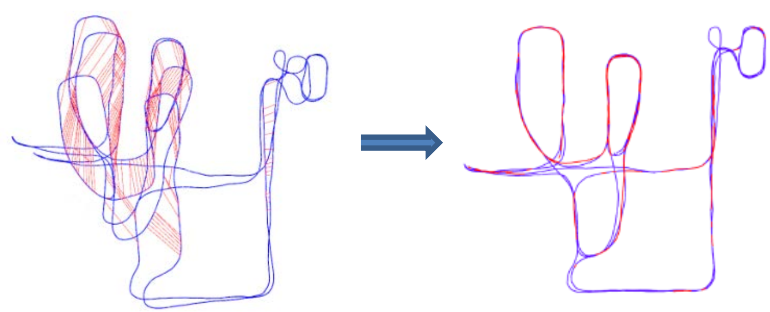
$$\min_{\mathbf{p},\psi} \left\{ \sum_{(i,j)\in\mathcal{S}} \|\mathbf{r}_{i,j}\|^2 + \sum_{(i,j)\in\mathcal{L}} h(\|\mathbf{r}_{i,j}\|) \right\}$$

Loop closure edges

5. Add Keyframe into Pose Graph

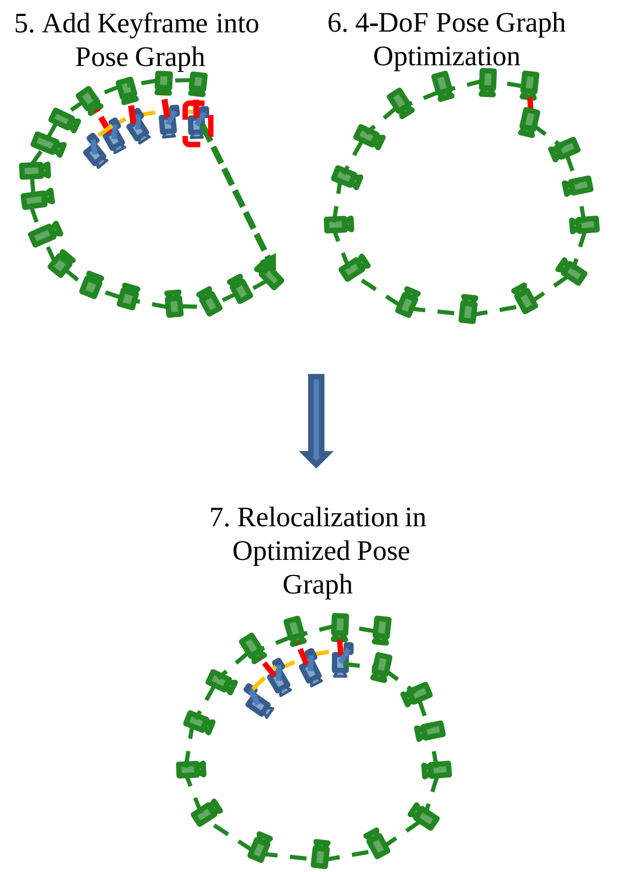
6. 4-DoF Pose Graph Optimization





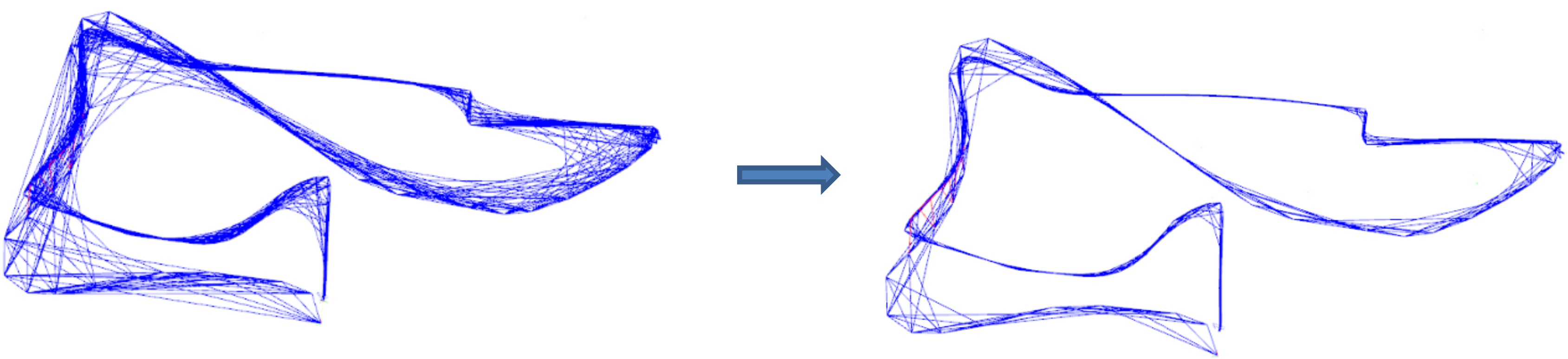
Global Pose Graph SLAM

- More on relocalization
  - Relocalization continued on the optimized pose graph
  - Relocalization and pose graph optimization run in different threads and in different rate
  - Pose graph optimization can be very slow for large-scale environments



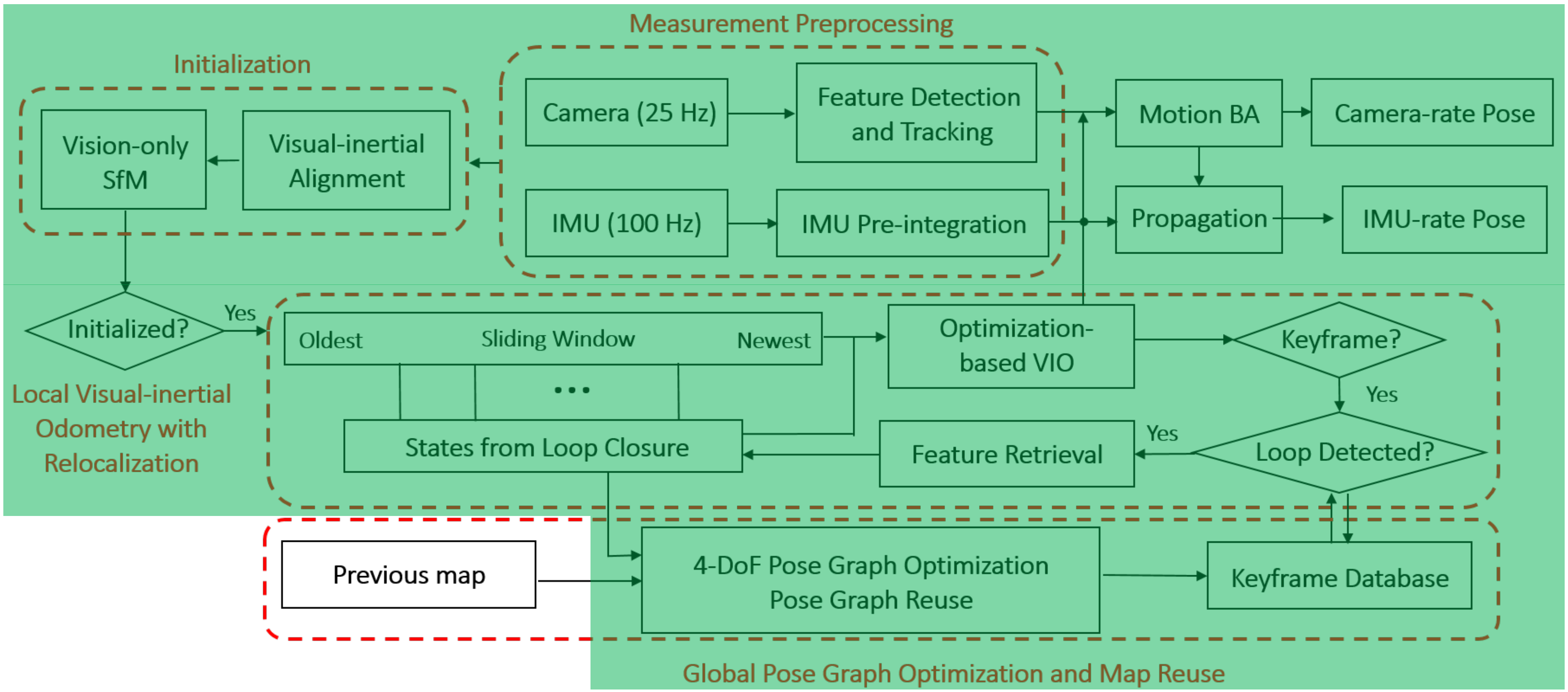
Global Pose Graph SLAM

- Simple strategy for pose graph sparsification
  - All keyframes with loop closure constraints will be kept
  - Other keyframes that are either too close to its neighbors or have very similar orientations will be removed



Monocular Visual-Inertial SLAM

- System diagram



Visual-Inertial SLAM in Large-Scale Environment

II. Go out laboratory

Single camera: mvBluefox
IMU: DJI A3

Pose Graph Reuse

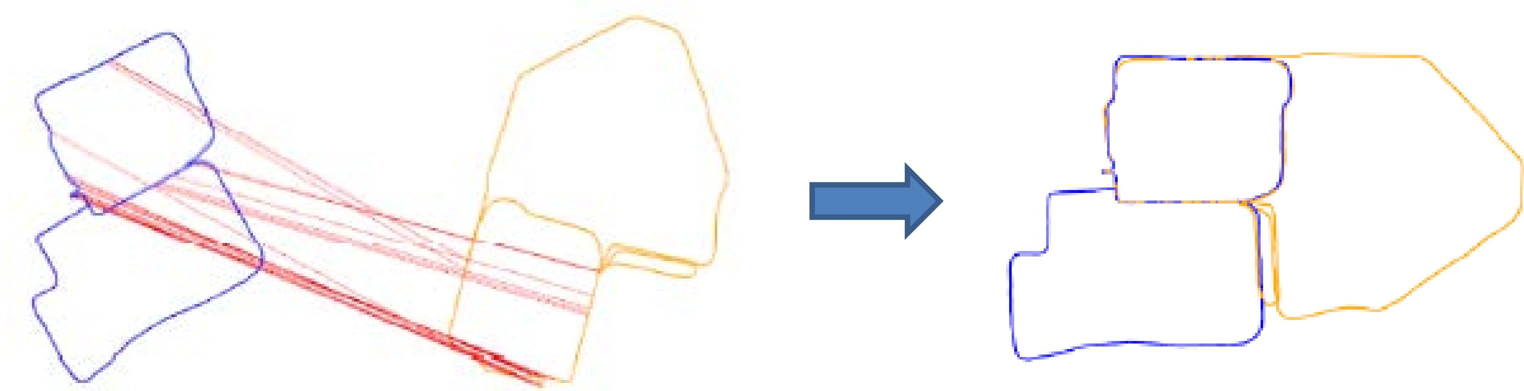
- Pose graph saving
  - Every Keyframe
    - Index i, position $\hat{\boldsymbol{p}}_i^w$, orientation $\hat{\boldsymbol{q}}_i^w$, features' 2D location and descriptor $D(u, v, des)$
    - If i loops with v, we also save loop index v, relative translation $\hat{p}_{iv}^i$, relative yaw angle $\hat{\varphi}_{iv}$

$$[i, \hat{\mathbf{p}}_i^w, \hat{\mathbf{q}}_i^w, v, \hat{\mathbf{p}}_{iv}^i, \hat{\psi}_{iv}, \mathbf{D}(u, v, des)]$$

- Pose graph loading
  - Build sequential edges
    - Connected with 4 previous keyframes
  - Build loop closure edges
    - According to loop index v, relative translation $\hat{\boldsymbol{p}}_{iv}^i$ and yaw angle $\hat{\varphi}_{iv}$

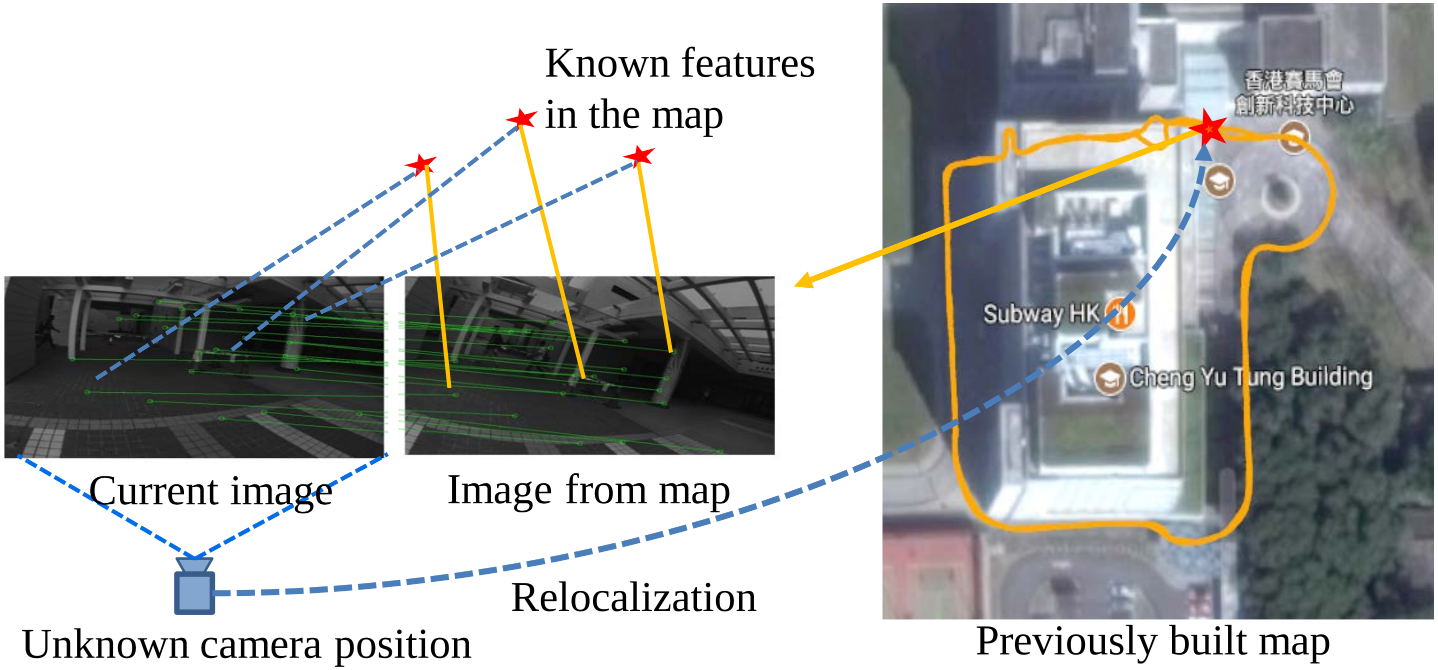
Pose Graph Reuse

- Pose graph merging
  - Load a previous-built map
  - Build a new map
  - Detect loop connections between two maps
  - Merge two map by pose graph optimization



Pose Graph Reuse

- Relocalization
  - Load previous-built map (aligned with Google Map)
  - The camera starts at an unknown position
  - Detect similar image view in the map
  - Once loop detected, relocate camera pose



香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY | 電子及計算機工程學系 DEPARTMENT OF ELECTRONIC & COMPUTER ENGINEERING

香港科大機器人研究所 ROBOTICS INSTITUTE

Pose Graph Reuse

Relocalization, Global Optimization and Map Merging for Monocular Visual-Inertial SLAM

Tong Qin, Peiliang Li, and Shaojie Shen

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

香港科技大學一大疆創新科技聯合實驗室 HKUST-DJI JOINT INNOVATION LABORATORY

Open source: https://github.com/HKUST-Aerial-Robotics/VINS-Mono

Remarks on Monocular Visual-Inertial SLAM

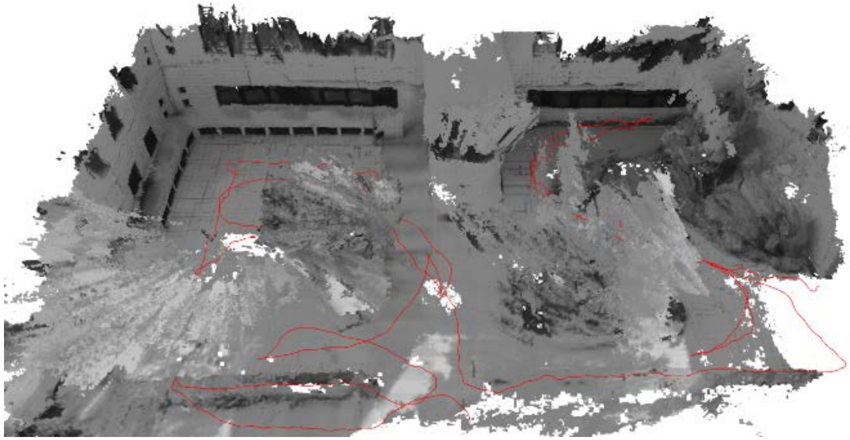
- Important factors
  - Access to raw camera data (especially for rolling shutter cameras)
  - Sensor synchronization and timestamps
  - Camera-IMU rotation
  - Estimator initialization
- Not-so-important factors
  - Camera-IMU translation
  - Types of features (we use the simplest corner+KLT)
  - Quality of feature tracking (outlier is acceptable)
- Failures – need more engineering treatment
  - Long range scenes (aerial vehicles)
  - Constant velocity (ground vehicle)
  - Pure rotation (augmented reality)
- Be aware of computational power requirement

Remarks on Monocular Visual-Inertial SLAM

- IMU is great!!!
- Feature-based visual-inertial SLAM is very close to done
  - Some research work remains:
    - Online observability analysis
    - Large-scale, long duration operations
    - Extreme environments
    - Extreme motions
  - Big engineering challenges towards mass deployment on different devices (Android phones?)
    - Intrinsic and extrinsic calibration of IMU, rolling shutter, etc.
    - Synchronization issues
    - Poor sensors and manufacturing variations
    - Insufficient computing power
  - Big players are moving in

Remarks on Monocular Visual-Inertial SLAM

- Real-time dense mapping is interesting
  - Very few working implementations
  - How to reduce computation?
  - Parallel implementation on GPU
  - Joint optimization or alternating estimation?
  - Textureless and repetitive patterns?
  - Combination of learning and geometric-based methods
  - Efficient map representation for large-scale environments



Dense Mapping, Trajectory Planning, and Navigation

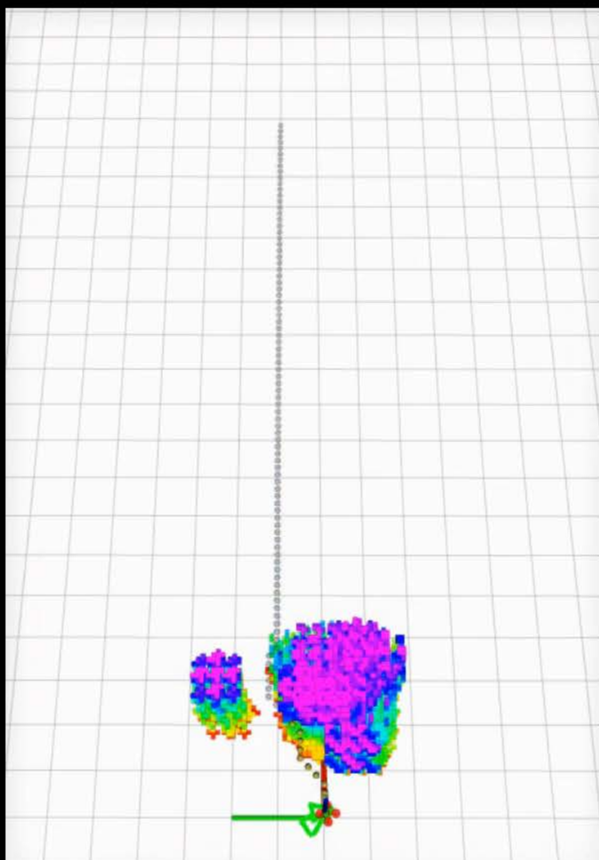
Indoor Experiment 2:

Trajectory length: 18.6m

Total number of replans: 125

Average computing time: 43ms

Average snap: 1.21m/s^4



2x

2x

Thanks!

Questions?