Monocular Visual-Inertial SLAM

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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Why Monocular?

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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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

Challenges: Monocular Vision

• Scale ambiguity

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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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

\[
\begin{align*}
v_0 &= \text{?} \\
g_0 &= \text{?} \\
R^b_c &= \text{?} \\
p^b_c &= \text{?}
\end{align*}
\]
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
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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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 integrate IMU measurements


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
The Bad of IMU Integration in the Global Frame

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

\[
\begin{align*}
p_{b_{k+1}}^w &= p_{b_k}^w + v_{b_k}^w \Delta t_k \\
&\quad + \int_{t \in [t_k, t_{k+1}]} R_{t_k}^w \left( \mathbf{a}_t - b_{ax} - n_{ax} - g^w \right) dt^2 \\
v_{b_{k+1}}^w &= v_{b_k}^w + \int_{t \in [t_k, t_{k+1}]} R_{t_k}^w \left( \mathbf{a}_t - b_{ax} - n_{ax} - g^w \right) dt \\
q_{b_{k+1}}^w &= q_{b_k}^w \times \int_{t \in [t_k, t_{k+1}]} \frac{1}{2} \mathbf{\Omega}(\mathbf{a}_t - b_{aw} - n_{aw}) q_{b_k}^b dt, \\
\mathbf{\Omega}(\omega) &= \begin{bmatrix} -[\omega]_x & \omega & 0 \\ -\omega^T_x & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}, \\
[\omega]_x &= \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}.
\end{align*}
\]

This Does Not Work!


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk

IMU body frame

World frame
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

\[
\begin{align*}
R_{b_{k+1}}^w p_{b_{k+1}} &= R_{b_k}^w (p_{b_k}^w + v_{b_k}^w \Delta t_k - \frac{1}{2} g^w \Delta t_k^2) + \alpha_{b_{k+1}}^b \\
R_{b_k}^w v_{b_{k+1}} &= R_{b_k}^w (v_{b_k}^w - g^w \Delta t_k) + \beta_{b_{k+1}}^b \\
q_{b_k}^w \otimes q_{b_{k+1}}^w &= \gamma_{b_{k+1}}^b,
\end{align*}
\]

\[
\begin{align*}
\alpha_{b_{k+1}}^b &= \int \int_{t\in[t_k,t_{k+1}]} R_t^{b_k} (\dot{a}_t - b_{a_t} - n_a) dt^2 \\
\beta_{b_{k+1}}^b &= \int \int_{t\in[t_k,t_{k+1}]} R_t^{b_k} (\dot{a}_t - b_{a_t} - n_a) dt \\
\gamma_{b_{k+1}}^b &= \int \int_{t\in[t_k,t_{k+1}]} \frac{1}{2} \Omega (\dot{\omega}_t - b_{\omega_t} - n_\omega) \gamma_t^{b_k} dt.
\end{align*}
\]


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
• Uncertainty propagation on manifold
  – Derive the error state model for the IMU pre-integration dynamics

\[
\begin{bmatrix}
\delta \alpha_{t}^{b_k} \\
\delta \beta_{t}^{b_k} \\
\delta \theta_{t}^{b_k} \\
\delta b_{a_t} \\
\delta b_{w_t}
\end{bmatrix}
= 
\begin{bmatrix}
0 & I & 0 \\
0 & 0 & -R_t^{b_k} [\hat{a}_t - b_{a_t}] \times \\
0 & 0 & -[\hat{\omega}_t - b_{w_t}] \times \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\delta \alpha_{t}^{b_k} \\
\delta \beta_{t}^{b_k} \\
\delta \theta_{t}^{b_k} \\
\delta b_{a_t} \\
\delta b_{w_t}
\end{bmatrix}
+ 
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & -R_t^{b_k} & 0 & 0 & 0 \\
0 & 0 & -I & 0 & 0 \\
0 & 0 & 0 & I & 0 \\
0 & 0 & 0 & 0 & I
\end{bmatrix}
\begin{bmatrix}
n_a \\
n_w \\
n_{b_a} \\
n_{b_w}
\end{bmatrix}
= F_t \delta z_{t_{b_k}}^{b_k} + G_t n_t.
\]

– Discrete-time implementation

\[
P_{t+\delta t}^{b_k} = (I + F_t \delta t) P_t^{b_k} (I + F_t \delta t)^T + \delta t G_t Q_t G_t^T,
\]
\[t \in [k, k+1].\]


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
• Jacobian matrices for bias correction
  – Also derive the Jacobian of the pre-integrated measurements w.r.t. IMU bias

\[
J_{b_k} = I, \quad J_{t+\delta t} = (I + F_t \delta t)J_t, \quad t \in [k, k+1]
\]

– And write down the linearized model for bias correction

\[
\begin{align*}
\alpha_{b_{k+1}} & \approx \alpha_{b_k} + J_{b_a} \delta b_{a_k} + J_{b_w} \delta b_{w_k} \\
\beta_{b_{k+1}} & \approx \beta_{b_k} + J_{b_a} \delta b_{a_k} + J_{b_w} \delta b_{w_k} \\
\gamma_{b_{k+1}} & \approx \gamma_{b_k} \otimes \left[ \frac{1}{2} J_{b_w} \delta b_{w_k} \right]
\end{align*}
\]
IMU Pre-Integration on Manifold

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

\[
\begin{bmatrix}
\hat{\alpha}_{b_{k+1}}
\hat{\beta}_{b_{k+1}}
\hat{\gamma}_{b_{k+1}}
0
0
\end{bmatrix}
= 
\begin{bmatrix}
R_w^b (p_{w_{k+1}} - p_{w_k} + \frac{1}{2} g^w \Delta t_k^2 - v_{b_{k}} \Delta t_k)
R_w^b (v_{w_{k+1}}^w + g^w \Delta t_k - v_{b_k}^w)
q_{b_{k}}^w \otimes q_{b_{k+1}}^w
b_{ab_{k+1}} - b_{ab_{k}}
b_{wb_{k+1}} - b_{wb_{k}}
\end{bmatrix}
\]
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:
    \[
    \begin{align*}
    \mathcal{X} &= \begin{bmatrix} x_0, x_1, \ldots, x_n, x^b_c, \lambda_0, \lambda_1, \ldots, \lambda_m \end{bmatrix} \\
    x_k &= \begin{bmatrix} p^w_{b_k}, v^w_{b_k}, q^w_{b_k}, b_a, b_g \end{bmatrix}, k \in [0, n] \\
    x^b_c &= \begin{bmatrix} p^b_c, q^b_c \end{bmatrix},
    \end{align*}
    \]

  - Minimize residuals from all sensors


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Monocular Visual-Inertial Odometry

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

$$
\mathbf{r}_B(\hat{x}_{b_{k+1}}, \mathcal{X}) = \begin{bmatrix}
\delta \alpha_{b_{k+1}}^b \\
\delta \beta_{b_{k+1}}^b \\
\delta \theta_{b_{k+1}}^b \\
\delta \omega_a^b \\
\delta \omega_g^b
\end{bmatrix}
$$


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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

\[
\begin{align*}
  r_C(z_i^c, X) &= \left( \begin{array}{c} \tilde{u}_i^c \\ \tilde{v}_i^c \end{array} \right) - \left( \begin{array}{c} u_i^c \\ v_i^c \end{array} \right) \\
  &= \left( \begin{array}{c} R_d^c (R_b^d (R_w^b (R_c^b \Sigma_i^{1/2} \pi_c^{-1} ([u_i^c, v_i^c]') + p_b^b) + p_b^w - p_b^w) - p_c^b \end{array} \right)
\end{align*}
\]


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Monocular Visual-Inertial Odometry

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

\[
\mathbf{r}_C(\hat{z}_l^c, \mathcal{X}) = \begin{bmatrix} b_1 & b_2 \end{bmatrix}^T \cdot \left( \bar{\mathbf{P}}_l^c - \frac{\mathbf{P}_l^c}{\|\mathbf{P}_l^c\|} \right)
\]

\[
\bar{\mathbf{P}}_l^c = \pi_c^{-1} \left( \begin{bmatrix} \hat{u}_l^c & \hat{v}_l^c \end{bmatrix} \right)
\]

\[
\mathbf{P}_l^c = R_b^c (R^w_b \mathbf{R}_b^w \mathbf{R}_c^b \mathbf{R}_c^{b_1} \lambda_l \pi_c^{-1} \left( \begin{bmatrix} u_l^{c_1} & v_l^{c_1} \end{bmatrix} \right) + \mathbf{p}_c^b) + \mathbf{p}_b^w - \mathbf{p}_b^{b_1} - \mathbf{p}_c^b
\]

Reprojection error on tangent plane

Unit vector of feature from 2D measurement

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 $\vec{P}_l^{c_j}$, e.g. [1 0 0]
    • $b_1 = normalize(\vec{P}_l^{c_j} \times [1 0 0])$
    • $b_2 = normalize(\vec{P}_l^{c_j} \times b_1)$

• Spherical vs. pinhole camera models
  – Different ways to define the reprojection error
  – Able to model cameras with arbitrary FOV


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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**

Monocular Visual-Inertial Odometry

- Vision measurement residual for temporal calibration
  - Feature velocity on image plane
    - feature \( l \) moves at speed \( V^k_l \) from image \( k \) to \( k + 1 \) in short time period \([t_k, t_{k+1}]\)
    \[
    V^k_l = (\begin{bmatrix} u_k^{k+1} \\ v_k^{k+1} \end{bmatrix} - \begin{bmatrix} u_k^k \\ v_k^k \end{bmatrix}) / (t_{k+1} - t_k)
    \]
  - Visual measurement residual with time offset
    - New state variable \( t_d \), and estimate states \((c_i', c_j')\) at time stamping

"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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Monocular Visual-Inertial Odometry

- Marginalization via Schur complement on information matrix

\[
\begin{bmatrix}
H_{mm} & H_{mr} \\
H_{rm} & H_{rr}
\end{bmatrix}
\begin{bmatrix}
x_m \\
x_r
\end{bmatrix} =
\begin{bmatrix}
r_m \\
r_r
\end{bmatrix}
\]

\[
(H_{rr} - H_{rm}H_{mm}^{-1}H_{mr})x_r = r_r - H_{rm}H_{mm}^{-1}r_m
\]


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Monocular Visual-Inertial Odometry

• Solving the nonlinear system
  – Minimize residuals from all sensors
    \[
    \min_{\mathbf{x}} \left\{ \| \mathbf{r}_p - \mathbf{H}_p \mathbf{x} \|^2 + \sum_{k \in B} \| \mathbf{r}_B(\hat{\mathbf{z}}_{bk+1}^b, \mathbf{x}) \|^2_{\mathbf{P}_{bk+1}^b} + \sum_{(i,j) \in C} \| \mathbf{r}_C(\hat{\mathbf{z}}_{ij}^c, \mathbf{x}) \|^2_{\mathbf{P}_{ij}^c} \right\}
    \]
  – 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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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
  \[ X = [x_0, x_1, \ldots, x_n, x^b, \lambda_0, \lambda_1, \ldots, \lambda_m] \]
  \[ x_k = [p^w_k, v^w_k, q^w_k, b_\alpha, b_\beta], \quad k \in [0, n] \]
  \[ b^c = [p^c, q^c] \]

- Prior in cost function is ignored

- Also solved using the Ceres Solver


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk

Old keyframes
Latest frames
IMU constraints
Features
Fixed states
Estimated states

IMU measurement residual
Vision measurement residual

Covariance from IMU pre-integration
Pixel reprojection covariance
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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Estimator Initialization

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

\[ \mathcal{X}_I = [v_{b_0}^{c_0}, v_{b_1}^{c_0}, \ldots, v_{b_n}^{c_0}, g^{c_0}, s] \]

---


@2018 HKUST Aerial Robotics Group | [http://uav.ust.hk](http://uav.ust.hk)
Estimator Initialization

- **Visual-inertial alignment**
  - Linear measurement model
    \[
    \hat{z}_{b_{k+1}}^{b_k} = \begin{bmatrix} \alpha_{b_{k+1}} \\ \beta_{b_{k+1}} \end{bmatrix} = R_{c_0}^{b_k} p_{b}^{c} + R_{c_0}^{b_k} p_{b}^{c} = H_{b_{k+1}}^{b_k} x_{I} + n_{b_{k+1}}^{b_k}
    \]
    Known values from vSfM and extrinsic calibration

- IMU Pre-integration

- Solve a linear system
  - Scale and rotate the vSfM
    \[
    x_{I} = [v_{b_0}^{c_0}, v_{b_1}^{c_0}, \cdots, v_{b_n}^{c_0}, g_{c_0}^{g}, s]
    \]
    Up-to-scale translation from vSfM

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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Monocular Visual-Inertial SLAM

- System diagram


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Visual-Inertial SLAM for Autonomous Drone

Monocular Visual-Inertial System (VINS-Mono) on MAV Platform for Autonomous Flight

Tong Qin, Peiliang Li, Zhenfei Yang and Shaojie Shen

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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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
Monocular Visual-Inertial SLAM

- **System diagram**


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Monocular Visual-Inertial Odometry with Relocalization

States in the sliding window
States from loop closure
IMU measurements
Visual measurements
Features

Loop closure frames with constant pose
Loop closure feature matches

@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Monocular Visual-Inertial Odometry with Relocalization

- **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

\[
\begin{align*}
\min_{\mathcal{X}} \left\{ & \| \mathbf{r}_p \|_{H_p}^2 + \sum_{k \in \mathcal{B}} \| \mathbf{r}_B(\hat{\mathbf{z}}_{b_{k+1}}^b, \mathcal{X}) \|_{P_{b_{k+1}}}^2 + \\
& \sum_{(l,j) \in \mathcal{C}} \| \mathbf{r}_C(\hat{\mathbf{z}}_{l}^c, \mathcal{X}) \|_{P_{l}^c}^2 + \sum_{(l,v) \in \mathcal{L}} \| \mathbf{r}_C(\hat{\mathbf{z}}_{v}^w, \mathcal{X}, \hat{\mathbf{q}}_{v}^w, \hat{\mathbf{P}}_{v}^w) \|_{P_{l}^w}^2 \right\}.
\end{align*}
\]


© 2018 HKUST Aerial Robotics Group | http://uav.ust.hk
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
  \[
  \hat{p}_{ij}^i = \hat{R}_i^{w} (\hat{p}_j^w - \hat{p}_i^w) \\
  \hat{\psi}_{ij} = \hat{\psi}_j - \hat{\psi}_i
  \]
  - 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
Global Pose Graph SLAM

• 4-DOF relative pose residual:

\[ r_{i,j}(p_i^w, \psi_i, p_j^w, \psi_j) = \begin{bmatrix} R(\phi_i, \theta_i, \psi_i)^{-1}(p_j^w - p_i^w) - \hat{p}_{ij} \\ \psi_j - \psi_i - \hat{\psi}_{ij} \end{bmatrix} \]

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

\[
\min_{p, \psi} \left\{ \sum_{(i,j) \in S} \| r_{i,j} \|^2 + \sum_{(i,j) \in L} h(\| r_{i,j} \|) \right\}
\]

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 $\hat{\mathbf{p}}^w_i$, orientation $\hat{\mathbf{q}}^w_i$, features’ 2D location and descriptor $D(u, v, des)$
    • If $i$ loops with $v$, we also save loop index $v$, relative translation $\hat{\mathbf{p}}^i_{iv}$, relative yaw angle $\hat{\psi}_{iv}$

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

• Pose graph loading
  – Build sequential edges
    • Connected with 4 previous keyframes
  – Build loop closure edges
    • According to loop index $v$, relative translation $\hat{\mathbf{p}}^i_{iv}$ and yaw angle $\hat{\psi}_{iv}$
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


@2018 HKUST Aerial Robotics Group | http://uav.ust.hk
Pose Graph Reuse

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

Tong Qin, Peiliang Li, and Shaojie Shen

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
Indoor Experiment 2:

Trajectory length: 18.6m
Total number of replans: 125
Average computing time: 43ms
Average snap: 1.21m/s

Y. Lin et al, JFR 2017; W. Ding et al, ICRA2018