Vision Algorithms for Mobile Robotics

Lecture 13
Visual Inertial Fusion

Davide Scaramuzza
http://rpg.ifi.uzh.ch
Today: Lab Exercise and Lab visit afterwards

Bundle Adjustment
Outline

• What is an IMU and why do we need it?
• IMU model
• Visual Inertial Odometry (VIO)
  • Closed-form solution
  • Non-linear optimization methods
  • Filtering methods
• Camera-IMU extrinsic calibration and Synchronization
What is an IMU?

• **Inertial Measurement Unit**
  • Gyroscope: Angular velocity
  • Accelerometer: Linear Accelerations

[Diagram of Inertial Measurement Unit]

Mechanical Gyroscope

Mechanical Accelerometer
What is an IMU?

• Different categories
  • Mechanical ($100,000-1M)
  • Optical ($20,000-100k)
  • MEMS (from 1$ (phones) to 1,000$
    (higher cost because they have a microchip
    running a Kalman filter)

• For small mobile robots & drones:
  MEMS IMU are mostly used
    • Cheap
    • Power efficient
    • Light weight and solid state
A spring-like structure connects the device to a seismic mass vibrating in a capacitive divider. A capacitive divider converts the displacement of the seismic mass into an electric signal. Damping is created by the gas sealed in the device.
MEMS Gyroscopes

- MEMS gyroscopes measure the **Coriolis forces** acting on MEMS vibrating structures (tuning forks, vibrating wheels, or **resonant solids**)
- Their working principle is similar to the haltere of a fly
- Haltere are small structures of some two-winged insects, such as flies. They are flapped rapidly and function as gyroscopes, informing the insect about rotation of the body during flight.
Why do we need an IMU?

• Monocular vision is **scale ambiguous** (Lecture 8, slide 7)
• Pure vision is **not robust enough**
  • Low texture
  • Under or over exposure (caused by low Dynamic Range)
  • Motion blur

Robustness is a critical issue: **Tesla accident, 2016:**

“The **autopilot sensors on the Model S failed to distinguish a white tractor-trailer crossing the highway against a bright sky.**” [The Guardian]
Why is an IMU alone not enough?

• Pure IMU integration will lead to large drift (especially in cheap IMUs)
• Example: 1D scenario. Double integration of acceleration returns the position:

\[ x(t) = x_0 + v_0(t - t_0) + \int_{t_0}^{t} a(\tau)d\tau^2 \]

• If there is a constant bias in the acceleration, the error of position will be proportional to \( t^2 \)
• Similarly for the orientation: if there is a bias in angular velocity, the error is proportional to the time \( t \)

Table from VectorNav, one of the best IMU companies. Errors were computed assuming the device at rest:
https://www.vectornav.com/resources/inertial-navigation-primer/specifications--and--error-budgets/specs-inserrorbudget

Automotive, smartphones, and drones accelerometers
Why visual inertial fusion?

IMU and vision are complementary

<table>
<thead>
<tr>
<th>Cameras</th>
<th>IMU</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exteroceptive sensor</strong>: measures light energy from the environment</td>
<td><strong>Proprioceptive sensor</strong>: measures values internal to the system</td>
</tr>
<tr>
<td>× Sensitive to motion blur, HDR, texture</td>
<td>✓ Insensitive to motion blur, HDR, texture</td>
</tr>
<tr>
<td>✓ Drift is bounded when observing the same scene</td>
<td>× Drift grows unbounded regardless of the environment</td>
</tr>
<tr>
<td>✓ Precise in slow motion</td>
<td>× Less precise in slow motion (low signal-to-noise ratio)</td>
</tr>
<tr>
<td>× Limited output rate (~100 Hz)</td>
<td>✓ High output rate (1,000-10,000 Hz)</td>
</tr>
<tr>
<td>× Scale ambiguity in monocular setup</td>
<td>✓ Can be used as a prior to predict next feature positions</td>
</tr>
</tbody>
</table>

What cameras and IMU have in common: both can be used to estimate the pose incrementally (known as dead-reckoning), which suffers from drift over time. **Solution: fuse them together to reduce drift (see later)**
Outline

• What is an IMU and why do we need it?
  • IMU model
• Visual Inertial Odometry (VIO)
  • Closed-form solution
  • Non-linear optimization methods
  • Filtering methods
• Camera-IMU extrinsic calibration and Synchronization
IMU Measurement Model

The model measures the **angular velocity** $\tilde{\omega}_B(t)$ and **acceleration** $\tilde{a}_B(t)$ **vectors** in the body frame $B$:

$$
\begin{align*}
\tilde{\omega}_B(t) &= \omega_B(t) + b^G(t) + n^G(t) \\
\tilde{a}_B(t) &= R_{BW}(t)(a_W(t) - g) + b^A(t) + n^A(t)
\end{align*}
$$

**Notation:**
- The superscript $(\cdot)^G$ stands for gyroscope and $(\cdot)^A$ for accelerometer
- $R_{BW}$ is the rotation of the World frame $W$ with respect to Body frame $B$
- The gravity vector $g$ is expressed in the World frame
- Biases and noise are expressed in the body frame
IMU Noise and Bias Model

- Additive, zero-mean Gaussian white noise: $n^G(t), n^A(t)$
- Biases: $b^G(t), b^A(t)$
  - The gyroscope and accelerometer biases are considered \textit{slowly varying “constants”}. Their temporal fluctuation is modeled by saying that the derivative of the bias is a zero-mean Gaussian noise with standard deviation $\sigma_b$

$$\dot{b}(t) = \sigma_b \mathbf{w}(t) \quad \mathbf{w}(t) \sim \mathbf{N}(0,1)$$

- Some facts about IMU biases:
  - They change with \textit{temperature} and \textit{mechanical and atmospheric pressure}
  - Thus, they \textbf{may also be different every time the IMU is turned on}
  - Good news: \textbf{they can be estimated!} (see later)


The IMU Integration Model computes the position, orientation, and velocity of the IMU in the world frame. To do this, we must first compute the acceleration $a(t)$ in the world frame from the measured one $\ddot{a}(t)$ in the body frame (see Slide 12):

$$a(t) = R_{WB}(t) \left( \ddot{a}(t) - b(t) \right) + g$$

The position $p_k$ at time $t_k$ can then be predicted from the position $p_{k-1}$ at time $t_{k-1}$ by integrating all the inertial measurements $\{\ddot{a}_j, \dddot{\omega}_j\}$ within that time interval:

$$p_k = p_{k-1} + v_{k-1}(t_k - t_{k-1}) + \int_{t_{k-1}}^{t_k} \left( R_{WB}(t) \left( \ddot{a}(t) - b(t) \right) + g \right) dt^2$$

NB:
- The rotation $R_{WB}$ is computed from the gyroscope
- $p_k$ depends on initial position and velocity. How do we measure them?

A similar expression can be obtained to predict the velocity $v_k$ and orientation $R_{WB}$ of the IMU in the world frame as functions of both $\ddot{a}_j$ and $\dddot{\omega}_j$.

IMU Integration Model

For convenience, the IMU Integration Model is normally written as

\[
\begin{pmatrix}
p_k \\
q_k \\
v_k \end{pmatrix} = f \begin{pmatrix}
p_{k-1} \\
q_{k-1} \\
v_{k-1} \\
u \end{pmatrix}
\]

or, more compactly:

\[
x_k = f(x_{k-1}, u)
\]

where:

- \( x = \begin{bmatrix} p \\ q \\ v \end{bmatrix} \) represents the IMU state, i.e., position, orientation, and velocity
- \( q \) is the IMU orientation \( R_{WB} \) (usually represented using quaternions)
- \( u = \{ \ddot{a}_j, \ddot{\omega}_j \} \) are the accelerometer and gyroscope measurements in the time interval \( [t_{k-1}, t_k] \)

Outline

• What is an IMU and why do we need it?

• IMU model

• Visual Inertial Odometry (VIO)
  • Closed-form solution
  • Non-linear optimization methods
  • Filtering methods

• Camera-IMU extrinsic calibration and Synchronization
Visual Inertial Odometry

Different paradigms exist:

- **Loosely coupled:**
  - Treats **VO and IMU as two separate black boxes** (not coupled)
    - Each black box estimates pose and velocity from visual (up to a scale) and inertial data (absolute scale)
    - Easy to implement
    - Inaccurate. Should not be used

- **Tightly coupled:**
  - Makes use of the **raw sensors’ measurements** (2D features and IMU readings)
    - More accurate
    - More implementation effort

In this lecture, we will only see **tightly coupled approaches**
The Loosely Coupled Approach

Feature tracking

IMU Integration

VO

Position (up to a scale) & Orientation
(3D landmarks are estimated by VO but not refined by IMU)

Fusion

2D features

Refined Position
Orientation
Velocity

Images

IMU measurements

Position Orientation Velocity
The **Tightly Coupled** Approach

- **Feature tracking**
- **IMU measurements**
- **2D features**
- **Fusion**
  - Refined Position
  - Orientation
  - Velocity
  - 3D landmarks
Filtering: Visual Inertial Formulation

• System states:

  **Tightly coupled**: \( X = [p(t); q(t); v(t); b^A(t); b^G(t); L_1; L_2; \ldots; L_k] \)

  **Loosely coupled**: \( X = [p(t); q(t); v(t); b^A(t); b^G(t)] \)

Outline

• What is an IMU and why do we need it?
• IMU model
• Visual Inertial Odometry (VIO)
  • Closed-form solution
  • Non-linear optimization methods
  • Filtering methods
• Camera-IMU extrinsic calibration and Synchronization
Closed-form Solution (1D case)

• From a **single camera** we only get the relative position $\tilde{x}$ up to an **unknown scale factor** $s$, thus the absolute position $x$ is:

$$x = s\tilde{x}$$

• From the IMU

$$x = x_0 + v_0(t_1 - t_0) + \int_{t_0}^{t_1} a(t)dt^2$$

• By equating them

$$s\tilde{x} = x_0 + v_0(t_1 - t_0) + \int_{t_0}^{t_1} a(t)dt^2$$

As shown in [Martinelli’14], if we assume to know $x_0$ (usually we set it to 0), then, for 6DOF, **both $s$ and $v_0$ can be determined in closed form** from a **single feature observation and 3 views**

Martinelli, *Closed-form solution of visual-inertial structure from motion*, International Journal of Computer Vision (IJCV), 2014. [PDF](#)
Outline

• What is an IMU and why do we need it?
• IMU model
• Visual Inertial Odometry (VIO)
  • Closed-form solution
  • Non-linear optimization methods
  • Filtering methods
• Camera-IMU extrinsic calibration and Synchronization
Non-linear Optimization Methods

VIO is solved as a non-linear Least Square optimization problem over:

$$\{X, L, b^A, b^G\} = \arg\min_{\{X, L, b^A, b^G\}} \left\{ \sum_{k=1}^{N} \| f(x_{k-1}, u) - x_{k} \|_{A_k}^2 + \sum_{k=1}^{N} \sum_{i=1}^{M} \| \pi(x_k, L_i) - z^i_k \|_{\Sigma_k}^2 \right\}$$

**IMU residuals**  
**Reprojection residuals**  
(Bundle Adjustment term)

NB: it also optimizes the biases

Which initial guess do we use for the state and the biases?


Non-linear Optimization Methods

VIO is solved as a non-linear Least Square optimization problem over:

\[
\{X, L, b^A, b^G\} = \arg\min_{X, L, b^A, b^G} \left\{ \sum_{k=1}^{N} \| f(x_{k-1}, u) - x_k \|_A^2 + \sum_{k=1}^{N} \sum_{i=1}^{M} \| \pi(x_k, L^i) - z_k^i \|_{\Sigma_k}^2 \right\}
\]

where

- \( X = \{x_1, ..., x_N\} \): set of state estimates \( x_k \) (position, velocity, orientation) at frame times \( k \)
- \( L = \{L_1, ..., LM\} \): 3D landmarks
- \( f(x_{k-1}, u) \): state prediction obtained by integrating IMU measurements \( u = \{\bar{a}_j, \bar{\omega}_j\} \)
- \( \pi(x_k, L^i) \): expected measurement at state estimates \( x_k \) from projection of landmark \( L^i \) onto camera frame \( I_k \)
- \( z_{ik} \): observed features
- \( \Lambda_k \): state covariance from the IMU integration \( f(x_{k-1}, u) \)
- \( \Sigma_{ik} \): covariance of the 2D feature position

Case Study 1: OKVIS

Because the complexity of the optimization is cubic with respect to the number of cameras poses and features, real-time operation becomes infeasible as the trajectory and the map grow over time, OKVIS proposed to only optimize the current pose and a window of past keyframes.

Case Study 2: SVO+GTSAM

Solves the same optimization problem as OKVIS but:

- **Keeps all the frames** (from the start to the end of the trajectory)
- **To make the optimization efficient**
  - Marginalizes 3D landmarks (minimizes Epipolar Line Distance (Lecture 08) instead of the reprojection error)
  - **pre-integrates the IMU** data between keyframes (see later)
- **Optimization solved using Factor Graphs** via GTSAM
  - Very fast because it only **optimizes the poses** that are **affected by a new observation**

SVO+GTSAM

SVO+GTSAM

Problem with IMU integration

• The integration of IMU measurements, $f(x_{k-1}, u)$, from $k - 1$ to $k$ is related to the state estimation at time $k - 1$

• During optimization, every time the linearization point at $k - 1$ changes, the integration between $k - 1$ and $k$ must be re-evaluated, thus slowing down the optimization

$$\{X, L, b^A, b^G\} = \text{argmin}_{\{X, L, b^A, b^G\}} \left\{ \sum_{k=1}^{N} \| f(x_{k-1}, u) - x_k \|^2_{A_k} + \sum_{k=1}^{N} \sum_{i=1}^{M} \| \pi(x_k, L^i) - z_k^i \|^2_{\Sigma^i_k} \right\}$$

**IMU residuals**

**Reprojection residuals**

(Bundle Adjustment term)

• **Idea: Preintegration**
  • defines relative motion increments, expressed in body frame, which are independent on the global position, orientation, and velocity at $k$ [1]
  • [2] uses this theory by leveraging the manifold structure of the rotation group $\text{SO}(3)$


**IMU Pre-Integration**

\[ f(x_{k-1}, u) - x_k \]

- **Standard:** Evaluate error in global frame:
  - \( e_R = \hat{R}(\hat{\omega}, R_{k-1})^T R_k \)
  - \( e_V = \hat{v}(\hat{\omega}, \hat{a}, v_{k-1}) - v_k \)
  - \( e_p = \hat{p}(\hat{\omega}, \hat{a}, p_{k-1}) - p_k \)

- **Preintegration:** Evaluate relative errors (i.e., in body frame):
  - \( e_R = \Delta \hat{R}^T \Delta R \)
  - \( e_V = \Delta \hat{v} - \Delta v \)
  - \( e_p = \Delta \hat{p} - \Delta p \)

**Prediction** \( \hat{\cdot} \) \hspace{1cm} **Estimate** \( \cdot \)

**Repeats integration** when previous state changes!

Preintegration of IMU deltas possible with **no initial condition required**.
Outline

• What is an IMU and why do we need it?
• IMU model
• Visual Inertial Odometry (VIO)
  • Closed-form solution
  • Non-linear optimization methods
    • Filtering methods
• Camera-IMU extrinsic calibration and Synchronization
# Non-linear optimization vs. Filtering

<table>
<thead>
<tr>
<th>Non-linear Optimization methods</th>
<th>Filtering methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimize a <strong>window</strong> of multiple states (or all the states) using non-linear Least-Squares optimization</td>
<td>Solve the same problem by <strong>running only one iteration of the optimization</strong> function (e.g., using Extended Kalman Filter (EKF))</td>
</tr>
</tbody>
</table>

- ✔ Multiple iterations (it re-linearizes at each iteration)
- ✔ Achieves the highest accuracy
- ✔ Slower
- ✗ One iteration only
- ✗ Sensitive to linearization point
- ✔ Fastest
Case Study 1: ROVIO

- EKF state: \( X = [p(t); q(t); v(t); b^a(t); b^g(t); L_1; L_2; \ldots; L_k] \)

- Basic idea:
  1. **Prediction step**: predicts next position, velocity, orientation, and features using IMU integration model
  2. **Measurement update**: refines state by leveraging visual constraint (ROVIO minimizes the photometric error between corresponding points (alternative would be the reprojection error))

ROVIO: Problems

- **Complexity** of the EKF grows *quadratically* in the number of estimated landmarks
  - Thus, **max 20 landmarks** are tracked to allow real-time operation

- **Only updates the most recent state**
Case Study 2: MSCKF

- **MSCKF (Multi-State Constraint Kalman Filter)** updates multiple past poses \(\{p_{C1}, q_{C1}, ..., p_{CN}, q_{CN}\}\) in addition to the current state \(\{p(t), q(t), v(t)\}\). State vector:

\[
X = [p(t); q(t); v(t); b^A(t); b^G(t); p_{C1}; q_{C1}; ...; p_{CN}; q_{CN}]
\]

- **Prediction step**: same as ROVIO

- **Measurement update**:
  - Differently from ROVIO,
    - **landmark positions** are **not added to the state vector**, thus can run very fast independently of the number of features
    - Visual constraint is obtained from the **Epipolar Line Distance** (Lecture 08)

- Used in spacecraft landing (**NASA/JPL Moon and Mars landing**), **DJI drones**, **Google ARCore**

- Released open source within the OpenVins project: [https://github.com/rpng/open_vins](https://github.com/rpng/open_vins)

---


MSCKF running in Google ARCore (former Google Tango)

Outline

• What is an IMU and why do we need it?
• IMU model
• Visual Inertial Odometry (VIO)
  • Closed-form solution
  • Non-linear optimization methods
  • Filtering methods
• Camera-IMU extrinsic calibration and Synchronization
Camera-IMU Calibration

- **Goal**: estimate the **rigid-body transformation** $T_{BC}$ and time **offset** $t_d$ between the **camera** and the **IMU** caused by communication delays and the internal sensor delays (introduced by filters and logic).

- **Assumptions**: Camera and IMU rigidly attached. Camera intrinsically calibrated.

- **Data**:
  - Image points from calibration pattern (checkerboard or QR board)
  - IMU measurements: accelerometer $\{a_k\}$ and gyroscope $\{\omega_k\}$
Kalibr Toolbox

- Code: https://github.com/ethz-asl/kalibr/wiki/camera-imu-calibration

Kalibr Toolbox

- Solves a non-linear Least Square optimization problem similar to that seen before but also optimizes over $T_{BC}, t_d$:

$$\{X, L, T_{BC}, t_d, b^a, b^g\} = \arg\min_{\{X, L, T_{BC}, t_d, b^a, b^g\}} \left\{ \sum_{k=1}^{N} \| f(x_{k-1}, u) - x_k \|_{A_k}^2 + \sum_{k=1}^{N} \sum_{i=1}^{M} \| \pi(x_k, L^i) - z_k^i \|^2_{\Sigma_k} \right\}$$

  - **IMU residuals**
  - Reprojection residuals (Bundle Adjustment term)

- Continuous-time modelling using splines for $X$

- Numerical solver: Levenberg-Marquardt

---

Latest and Greatest 😊
• IMU-only odometry for pedestrians combining deep learning with an extended Kalman filter (EKF)
• A neural network regresses 3D displacement estimates and its uncertainty from a window of the most recent IMU measurements
• These displacements are fused into an EKF to estimate the pose, velocity and bias of the IMU.
• Enables robust state estimation in challenging environments for visual frontends, e.g. high dynamic scenes, low light, etc.

Readings

• Scaramuzza, Zhang, **Visual-Inertial Odometry of Aerial Robots**, Encyclopedia of Robotics, Springer, 2019, [PDF](#).

• Huang, **Visual-inertial navigation: A concise review**, International conference on Robotics and Automation (ICRA), 2019. [PDF](#).
Understanding Check

Are you able to answer the following questions?

• Why is it recommended to use an IMU for Visual Odometry?
• Why not just using an IMU and do inertial odometry (i.e., without a camera)?
• What is the basic idea behind MEMS IMUs?
• What is the drift of a consumer IMU?
• What is the IMU measurement model? (formula)
• What causes the bias in an IMU?
• How do we model the bias?
• How do we integrate the acceleration to get the position (formula)?
• What is the definition of loosely coupled and tightly coupled visual inertial fusion?
• How does non-linear optimization-based visual inertial odometry? Can you write down the cost function and illustrate its meaning?
• What does IMU-camera calibration do? Can you illustrate the unknowns and how to estimate them?