Inertial Navigation and Various Applications of Inertial Data

Yongcai Wang

9 November 2016
Types of Gyroscope

- Mechanical Gyroscope
- Laser Gyroscope
  Sagnac Effect
In **Stable platform**, the inertial sensors are mounted on a platform which is isolated from any external rotational motion, returning readings in **global frame**.

In **strapdown systems** the inertial sensors are mounted rigidly onto the device, returning readings in **body frame**.
Types of Gyroscopes: MEMS

Coriolis force

\[ F = 2mv \times w \]
## Accuracy Comparison

<table>
<thead>
<tr>
<th></th>
<th>GG1320AN (Laser)</th>
<th>MPU6050 (MEMS)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Size</strong></td>
<td>88<em>88</em>45</td>
<td>5<em>5</em>5</td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td>1.6Watt</td>
<td>5V, 3.6mA</td>
</tr>
<tr>
<td><strong>Angular random walk</strong></td>
<td>0.0035°/√h</td>
<td>&lt;0.2°/√h</td>
</tr>
<tr>
<td><strong>Bias stability</strong></td>
<td>0.0035°/√h</td>
<td>&lt;70°/√h</td>
</tr>
<tr>
<td><strong>Price</strong></td>
<td>&gt;5000</td>
<td>20</td>
</tr>
</tbody>
</table>
Inertial Navigation System
Inertial Navigation Systems

- Gyroscope provides angle velocity in body frame
  \[ \omega = (\omega_x, \omega_y, \omega_z) \]
- Accelerometer provides acceleration in body frame
  \[ a = (a_x, a_y, a_z) \]

1. Pose Update
2. Calculate linear ACC
3. Update velocity
4. Update position

Gravity \( g \)
Outline

- Pose update method
- Velocity and position update
- Error correction
- Step segmentation
- Pedestrian dead reckoning
Coordinate Frame Rotating

\[ R(t)R^T(t) = 1 \]
\[ \Rightarrow \dot{R}(t)R^T(t) + R(t)\dot{R}^T(t) = 0 \]

\[ S(t) = \dot{R}(t)R^T(t) \]
\[ \Rightarrow S(t) = -S^T(t) \Rightarrow \dot{R}(t) = S(t)R(t) \]

Angular velocity: \( \omega = (\omega_x, \omega_y, \omega_z) \)

The derivative of a time-varying rotation matrix:

\[ \dot{R}(t) = S(\omega)R(t) \]

\[
S(\omega) = \begin{pmatrix}
0 & -w_z & w_y \\
w_z & 0 & -w_x \\
-w_y & w_x & 0
\end{pmatrix}
\]
Coordinate Frame Rotating

• By first order Taylor explanation

\[
R(t + \delta_t) \approx \delta_t \dot{R}(t) + R(t)
\]
\[
\approx \delta_t S(\omega) R(t) + R(t)
\]
\[
\approx \left( \delta_t S(\omega) + I_{3 \times 3} \right) R(t)
\]

• Derivate the angular change from the rotation matrix

\[
\delta_t S(\omega) = R(t + \delta_t) R(t)^T - I_{3 \times 3}
\]
\[
\delta_\Theta = \delta_t \omega
\]
\[
= \text{vex}\left( \delta_t S(\omega) \right)
\]
\[
= \text{vex}\left( R(t + \delta_t) R(t)^T - I_{3 \times 3} \right)
\]
Continuous Rotation

✍️ Infinitesimal Rotating is Commuteive

\[ \mathbf{R}_{xyz} = \text{rotx}(0.001) \ast \text{roty}(0.002) \ast \text{rotz}(0.003) \]
\[ \mathbf{R}_{yxz} = \text{roty}(0.002) \ast \text{rotx}(0.002) \ast \text{rotz}(0.003) \]
\[ \mathbf{R}_{xyz} = \mathbf{R}_{yxz} \]

✍️ Recovery of the infinitesimal rotation angle \( \delta \Theta \)

\[ \text{vex}(\mathbf{R}_{xyz}) = \text{vex}(\mathbf{R}_{yxz}) = (0.001, 0.002, 0.003) \]

✍️ Difference between two infinitesimally poses

\[ \delta = \Delta \left( \xi_0, \xi_1 \right) = \left( \delta_d, \delta_\Theta \right) \]
\( \delta_d \) Incremental displacement
\( \delta_\Theta \) Incremental rotation
1. Pose update (1)

1. Pose update by first order Taylor approximation:

\[ R(k+1) = \delta_t S(\omega) R(k) + R(k) \]

2. Pose update by higher order approximation:

\[ \dot{R}(t) = S(\omega) R(t) \]

\[ \Rightarrow R(t) = R(0) \cdot \exp\left( \int_0^t \Omega(t) \, dt \right) \]

Let \( \int_{t-\delta_t}^t \Omega(t) \, dt = B_t \)

When \( \delta_t \) is small:

\[
B_t = \begin{pmatrix}
0 & -\omega^z \delta t & \omega^y \delta t \\
\omega^z \delta t & 0 & -\omega^x \delta t \\
-\omega^y \delta t & \omega^x \delta t & 0
\end{pmatrix}
\]

\[ R(t) = R(t-\delta_t) \cdot \exp\left( B_t \right) \]
1. Pose update (2)

\[ R(t) = R(t - \delta t) \cdot \exp \left( B_t \right) \]

\[ = R(t - \delta t) \left( I + \frac{B_t^2}{2!} + \frac{B_t^3}{3!} + \frac{B_t^4}{4!} + \cdots \right) \]

Let \( \sigma = |\omega \cdot \delta t| \), it is easy to verify that \( B_t^2 = -\sigma^2 \)

\[ R(t) = R(t - \delta t) \left( I + \frac{B_t^2}{2!} + \frac{B_t^3}{3!} + \frac{B_t^4}{4!} + \cdots \right) \]

\[ = R(t - \delta t) \left( I + \frac{B_t^2}{2!} - \sigma^2 B_t - \frac{\sigma^2 B_t^2}{4!} + \cdots \right) \]

\[ = R(t - \delta t) \left( I + \left( 1 - \frac{\sigma^2}{3!} + \frac{\sigma^4}{5!} + \cdots \right) B_t + \left( \frac{1}{2!} - \frac{\sigma^2}{4!} + \frac{\sigma^4}{5!} + \cdots \right) B_t^2 \right) \]

\[ = R(t - \delta t) \left( I + \frac{\sin \sigma}{\sigma} B_t + \frac{1 - \cos \sigma}{\sigma^2} B_t^2 \right) \]
Orthogonalize and Normalization

*R(t)* may become not orthonormal

*If the added term is small, the result will be close to orthonormal and we can straighten it up*

\[
\begin{align*}
  c_3' &= c_3 \\
  c_1' &= c_2' \times c_3' \quad \text{Normalization should be carried out after each step of integration} \\
  c_2' &= c_1' \times c_3' \\
  c_i'' &= \frac{c_i'}{|c_i'|}
\end{align*}
\]
2. Pose Update by Quaternions

- It becomes easier when using unit-quaternions.

- The derivative of $q$:

$$\dot{q} = \frac{1}{2} q(w) q,$$

where $q(w) = 0, <w>$.

- Integration of quaternion rate:

$$q(k+1) = \delta_t \dot{q} + q(k)$$

- Normalization

$$q'(k+1) = \frac{q(k+1)}{|q(k+1)|}$$
3. Velocity and Location Update

Convert the acceleration to global frame

\[ a^g_t = R(t) a^b_t \]

Exclude the gravity

\[ a^g_t = a^g_t - g^g \]

Double integral to calculate velocity and position

\[ v^g_t = v^g_{t-\delta t} + \delta t \cdot a^g_t \]

\[ p^g_t = p^g_{t-\delta t} + \delta t \cdot v^g_t \]
# MEMS gyroscope error sources

<table>
<thead>
<tr>
<th>Error type</th>
<th>Description</th>
<th>Result of single integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bias</td>
<td>A constant bias offset</td>
<td>A linearly growing error in angle</td>
</tr>
<tr>
<td>Random noise</td>
<td>Random noise from a variety of sources (usually specified in terms of its root PSD)</td>
<td>An angle random walk whose standard deviation grows proportional to the square root of time</td>
</tr>
<tr>
<td>Calibration</td>
<td>Deterministic errors in scale factors, alignments and gyro linearities</td>
<td>Orientation drift proportional to the rate and duration of motion</td>
</tr>
<tr>
<td>Temperature effects</td>
<td>Temperature dependent residual bias</td>
<td>Any residual bias is integrated into the orientation, causing an orientation error which grows linearly with time</td>
</tr>
<tr>
<td>Bias instability</td>
<td>Bias fluctuations, usually modelled as a bias random walk</td>
<td>A second-order random walk</td>
</tr>
</tbody>
</table>

\[ e_t = N(0,N(0,t)) + N(0,t) + c + N(0,Tt) \]

- bias
- Random noise
- calibration
- Temperature effects
INS Error Correction

- Using external measurements to correct the states of IMU.
  - Zero Velocity Update (ZUPT)
  - GPS, Other locating system

- Using a Kalman Filter to correct an INS based on external measurements.
  - Tracking full state of INS (openshoe)
  - Tracking state error of INS
INS Error Correction

- The true state of an IMU at time $t$

$$[R_t, v_t, p_t]$$

- The solution calculated by INS

$$[\hat{R}_t, \hat{v}_t, \hat{p}_t]$$

- The error state

$$\delta x_t = \begin{bmatrix} \delta \Phi_t^T, \delta v_t^T, \delta p_t^T \end{bmatrix}^T$$

in which

$$\delta \Phi_t = \begin{bmatrix} \delta \phi_t^{gx}, \delta \phi_t^{gy}, \delta \phi_t^{gz} \end{bmatrix}$$

- Relation of Error state and IMU State:

$$R_t = \Phi_t \hat{R}_t$$

$$v_t = \hat{v}_t + \delta v_t$$

$$p_t = \hat{p}_t + \delta p_t$$
INS Error Correction by KF

Estimated State Prediction

Error State Correction

External measurement

\[ \delta x_t = (\delta \Phi^T_t, \delta v^T_t, \delta p^T_t)^T \]

\[ R_t = \Phi_t \hat{R}_t \]
\[ v_t = \hat{v}_t + \delta v_t \]
\[ p_t = \hat{p}_t + \delta p_t \]

Reset \( \delta x_t \) to zero

INS System

\( (\hat{R}_t, \hat{v}_t, \hat{p}_t) \)

Working in parallel, without much changing to the INS system
Error State Prediction

State transition matrix

\[ F_t = \begin{pmatrix} I & 0 & 0 \\ \delta_t S(a_t^g) & I & 0 \\ 0 & \delta_t I & I \end{pmatrix} \]

where

\[ S(a_t^g) = \begin{pmatrix} 0 & -a_t^{gz} & a_t^{gz} \\ a_t^{gz} & 0 & -a_t^{gx} \\ -a_t^{gy} & a_t^{gx} & 0 \end{pmatrix} \]

Error State Prediction

\[ \delta \hat{\chi}_t^- = F_t \delta \hat{\chi}_{t-\delta t}^- \]

\[ P_t^- = F_t P_{t-\delta t} F_t^T + Q_t \]

Error State Prediction

\[ Q_t = \begin{pmatrix} T_t & 0 & 0 \\ 0 & U_t & 0 \\ 0 & 0 & 0 \end{pmatrix} \]

The noise item in state prediction

\[ T_t = \delta t^2 \left( \widehat{R}_t^- \right)^T W_t \left( \widehat{R}_t^- \right) \]

Representing noises of gyroscope

\[ W_t = \begin{pmatrix} (\sigma_w^{bx})^2 & 0 & 0 \\ 0 & (\sigma_w^{by})^2 & 0 \\ 0 & 0 & (\sigma_w^{bz})^2 \end{pmatrix} \]

\[ U_t = \delta t^2 \left( \widehat{R}_t^- \right)^T A_t \left( \widehat{R}_t^- \right) \]

Representing noises of accelerometer

\[ A_t = \begin{pmatrix} (\sigma_a^{bx})^2 & 0 & 0 \\ 0 & (\sigma_a^{by})^2 & 0 \\ 0 & 0 & (\sigma_a^{bz})^2 \end{pmatrix} \]
Error State Correction

- External measurement
- ZUPT
- Ultrasound…

\[ z_t = h_t(x_t) + v_t \]

In a simple test by keeping the IMU on the table,

\[ h_t(x_t) = v_t \]

![Graph showing mean drift over time](image)
KF-based correction vs. naïve correction
Pedestrian Dead Reckoning

Zero Velocity detection: ZUPT

\[ |\omega^b_t - \hat{\alpha}_t| < 0.5 \text{ rad/s} \]

\[ \delta x_t = (\delta \Phi^T_t, \delta \dot{v}_t, \delta p_t)^T \]

Reset \( \delta x_t \) to zero

\( R_t = \Phi_t \hat{R}_t \)
\( v_t = \dot{v}_t + \delta v_t \)
\( p_t = \ddot{p}_t + \delta p_t \)

INS System \( (\hat{R}_t, \hat{v}_t, \hat{p}_t) \)
Step Segmentation

- In practice, few systems or applications require updates at such a high frequency.

- Step events need to be detected for collaborating with other filtering algorithms.

- Parameterize a step event by: 
  \[ e_i = (l_t, \delta z_t, \delta \theta_t, \xi_t) \]
Step Segmentation: PDR filter

Step events are generated at the end of each stance phase.

1: **procedure** GENERATESTEP( (\( \hat{C}_t, \hat{s}_t, \hat{v}_t \)), (\( \hat{C}_{t-\delta t}, \hat{s}_{t-\delta t}, \hat{v}_{t-\delta t} \)) )

   // Calculate the change in displacement

2: \( \delta \hat{s} \leftarrow \hat{s}_t - \hat{s}_{t-\delta t} \)

   // Assume that the user’s heading (h) is aligned with the x-axis of the IMU

3: \( h^b \leftarrow (1, 0, 0)^T \)

   // Calculate h in the local frame at the start (\( h^g_{t-\delta t} \)) and end (\( h^g_t \)) of the step

4: \( h^g_{t-\delta t} \leftarrow \hat{C}_{t-\delta t} h^b \)

5: \( h^g_t \leftarrow \hat{C}_t h^b \)

   // Calculate the change in the user’s heading

6: \( \delta \theta_t \leftarrow \text{atan2}(h^g_t[1], h^g_t[0]) - \text{atan2}(h^g_{t-\delta t}[1], h^g_{t-\delta t}[0]) \)

   // Calculate the step offset

7: \( \xi_t \leftarrow \text{atan2}(h^g_t[1], h^g_t[0]) - \text{atan2}(\delta \hat{s}[1], \delta \hat{s}[0]) \)

   // Calculate the step length and vertical displacement

8: \( l_t \leftarrow \sqrt{\delta \hat{s}[0] \times \delta \hat{s}[0] + \delta \hat{s}[1] \times \delta \hat{s}[1]} \)

9: \( \delta z_t \leftarrow \delta \hat{s}[2] \)

   // Return the step event

10: **return** \( (l_t, \delta z_t, \delta \theta_t, \xi_t) \)

11: **end procedure**
Some Evaluation Results in [2]
2.5D view
Other Related works

- Using particle filters to correct PDR using map constraints [Woodman 2010].
- Using magnetometers to correct PDR [Zhou 2014, Mobicom]
- Using WiFi radiomap to correct PDR [Miyazaki 2014, Ubicomp]
Other cognitive problems using IMU

- Jogging
- Vacuuming
- Cooking
- Walking on stairs
- Brushing Teeth
- Blow Drying hair
- Practicing yoga
- Working on computer
- Dinning at restaurant
- Shopping
<table>
<thead>
<tr>
<th>Complexity</th>
<th>Category</th>
<th>Activity Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simple Activity</td>
<td>Walking, Jogging, Sitting, Standing, Lying, Walking Upstairs, Walking Downstairs, Jumping, Taking escalator up, Taking escalator down, Taking elevator up, Taking elevator down</td>
</tr>
<tr>
<td></td>
<td>Complex Activity</td>
<td>Shopping, Taking buses, Driving a car</td>
</tr>
<tr>
<td>Scenario</td>
<td>Living Activity</td>
<td>Brushing Teeth, Vacuuming, Eating, Cooking, Washing hands, Meditation, Clapping hands, Watering plants, Sweeping, Shaving, Blow drying hair, Washing dishes, Ironing, Flushing the toilet, Cleaning</td>
</tr>
<tr>
<td></td>
<td>Working Activity</td>
<td>Working on PC, Talking on the phone, On a break, Meeting, Typing, Writing, Doing a presentation</td>
</tr>
<tr>
<td></td>
<td>Health Activity</td>
<td>Exercising, Fall, Rehabilitation activities, Following routines</td>
</tr>
</tbody>
</table>
Process of Activity Recognition

1. Training Data Collection
2. Data Streaming
3. Data Processing
4. Classification Model
5. Test Data Streaming
6. Activity Recognition
7. Recognized Activity
Data collection

- Locations
  - Single location
  - Multiple location
Preprocessing

- De-noising
  - Signal Filter
    - High Pass Filter
    - Low Pass Filter

- Average
  - Acceleration Before Smoothing
  - Acceleration after Smoothing
Time domain Feature Extraction

- Time-Domain Features
  - Mean
  - Max, Min
  - Standard Deviation, Variance
  - Signal-Magnitude Area [8]
    \[ \sum_{i=1}^{n} \sqrt{x_i^2 + y_i^2 + z_i^2} \]
  - Correlation:
    \[ \frac{\text{cov}(x,y)}{\sigma_x \sigma_y} \]
Frequency Domain Features

- Energy: $\sum_{i=0}^{n} \sqrt{A_{\text{real}}_{i}^2 + A_{\text{image}}_{i}^2}$, $n$ is the window’s size. [9]
- Entropy: Normalized information entropy of the discrete FFT component [10]
- Time Between Peaks [11]
- Binned Distribution: histogram of FFTs [11]
Classification

- Base-level Classification
- Meta-level Classification
Base level classifier

- Decision Tree
- Decision Table
- K-Nearest Neighbor (KNN)
- Hidden Markov Model (HMM)
- Support Vector Machine (SVM)
- Naïve Bayes, Artificial Neural Networks, Gaussian Mixture Model, etc
Meta-level classifier

- **Voting**
  - Each base level classifier gives a vote, the class label receiving the most votes is the final decision.

- **Stacking**
  - Learning algorithm to learn how to combine the predictions of the base-level classifiers.

- **Cascading**
  - Iterative process to combine base-level classifiers.
    - Sub-optimal compared to others
Applications

- Calorie consumption estimation
- Fall detection
- Daily life logging
- Sport training
- Gait analysis for healthcare
Cognitive problems using IMU

Arm tracking by smart watch

5DOF model of Arm

[3] I am a Smartwatch and I can Track my User’s S. Shen, Mobicom 2016
Arm tracking by smart watch

Point cloud model of elbow and whist
1. Develop an PDR filter (Step Segmentation) based on the openshoe project.

http://www.openshoe.org/?page_id=362

- Compare the tracking accuracy of Step-based and INS-based
- Test the algorithms using self-collected IMU data traces.