



武汉大学测绘学院  
SCHOOL OF GEODESY AND GEOMATICS

课程编号：3150550011039

课程性质：必修

## 《惯性导航原理》实验报告

### 实验3: 纯惯导动态导航定位

学 院:	测绘学院
专 业:	智能导航
姓 名:	明佳浩
学 号:	2023302143030
指导老师:	朱峰、张万威、孙骁

二〇二五年十二月

## 目 录

第 1 章 实验概述 .....	1
1.1 实验目的 .....	1
1.2 完成情况 .....	1
第 2 章 实验原理与算法实现 .....	2
2.1 地球椭球相关常数计算原理 .....	2
2.1.1 地球几何常数与椭球模型 .....	2
2.1.2 正常重力模型 .....	3
2.1.3 地球自转角速度向量与位移角速度向量 .....	4
2.2 姿态表达式的相互转换 .....	5
2.2.1 欧拉角转方向余弦矩阵 .....	5
2.2.2 欧拉角转四元数 .....	5
2.2.3 方向余弦矩阵转欧拉角 .....	5
2.2.4 方向余弦矩阵转四元数 .....	6
2.2.5 四元数转方向余弦矩阵 .....	6
2.2.6 四元数转等效旋转矢量 .....	6
2.2.7 等效旋转矢量转方向余弦矩阵 .....	7
2.2.8 等效旋转矢量转四元数 .....	7
2.3 惯导机械编排算法 .....	8
2.3.1 速度更新算法 .....	8
2.3.2 位置更新算法 .....	11
2.3.3 姿态更新 .....	12
2.4 零速判断原理 .....	14
第 3 章 实验内容与步骤 .....	15
3.1 外业数据采集 .....	15
3.1.1 实验设备 .....	15
3.1.2 数据采集流程 .....	15
3.1.3 实验信息说明 .....	16
3.2 内业数据处理 .....	17
3.2.1 初始化参数 .....	18

3.2.2 数据读取 .....	18
3.2.3 数据补偿 .....	19
3.2.4 纯惯导解算 .....	20
3.2.5 其他说明 .....	20
<b>第 4 章 结果和分析 .....</b>	<b>21</b>
4.1 示例数据结果分析 .....	21
4.2 实测数据结果分析 .....	24
4.3 零速修正结果分析 .....	27
<b>第 5 章 总结 .....</b>	<b>30</b>
5.1 思考题 .....	30
5.2 问题与思考 .....	31
5.3 总结 .....	31
参考文献 .....	32

## 1.1 实验目的

- 1) 通过编程实现高精度惯导数据处理，加深对惯性导航原理和机械编排算法的理解；
- 2) 通过分析纯惯导解算的位置、速度和姿态的误差，对惯性导航的误差积累形成直观认识；
- 3) 初步掌握零速修正(将速度重置为零)，分析和理解零速修正对惯性导航误差的修正作用。

## 1.2 完成情况

本次实验总体流程如下图所示。

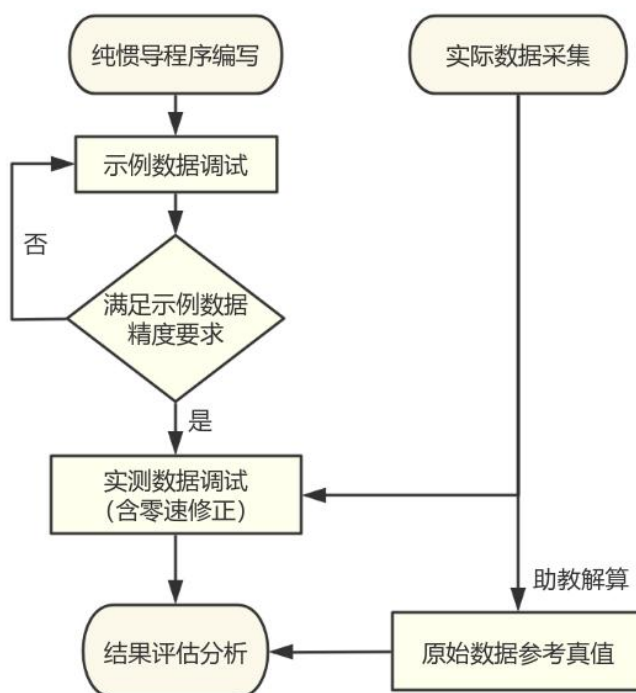


图 1.1 实验总体流程设计图

根据实验总体流程，本次实验集成了实验 1 的 IMU 误差标定和实验 2 静态粗对准演示示例，实现了基于示例数据和实测数据以及零速修正情况下的纯惯导导航定位程序，并基于 matlab 完成了结果绘制与精度评定程序。

实际运行过程中可以根据如下命令行参数有选择性地运行示例程序。

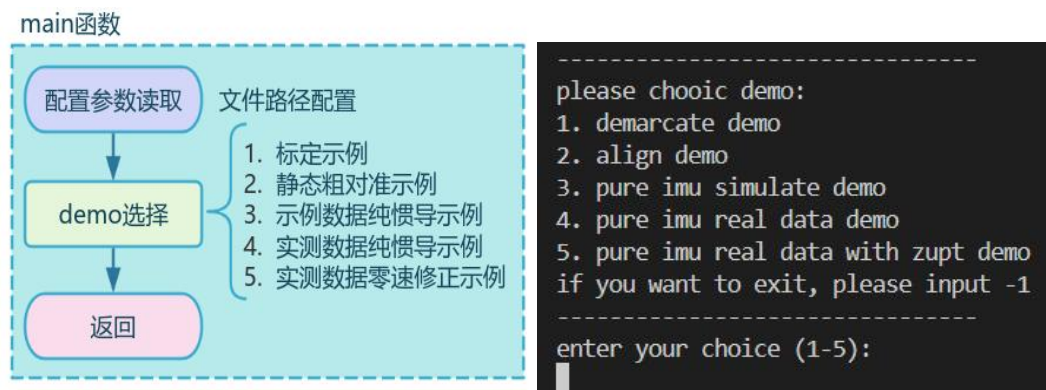


图 1.2 解算程序示例

## 第 2 章 实验原理与算法实现

惯性导航本质上是一种基于牛顿运动学的自主式航位推算技术。它不仅依赖于惯性测量单元(IMU)高频感知的载体比力与角速度等实时运动信息，还深度依赖于地球自转角速度、地球重力等外部物理常数。因此，本章节将在系统介绍惯性导航在导航坐标系(n 系)下的机械编排原理基础上，进一步介绍程序中所采用的地球常数模型、重力模型以及姿态变换相关的原理，从而为后续算法设计与系统实现奠定理论依据。

### 2.1 地球椭球相关常数计算原理

#### 2.1.1 地球几何常数与椭球模型

本次实验采取 GRS80 地球椭球模型，其对应地球几何常数如下：

表 2.1 GRS80 地球几何常数

地球几何常数	数值
长半轴 $a(m)$	6378137.0
短半轴 $b(m)$	6356752.3141
地球自转角速度 $\omega_e(rad/s)$	$7.292115 * 1e-5$
常数 $GM(m^3 / s^2)$	$3.986005 * 1e14$

根据所定义的地球椭球框架，可以计算得到目标纬度下的子午圈半径与卯酉圈半径

$$R_m = \frac{a(1-e^2)}{\sqrt{(1-e^2 \sin^2 \varphi)^3}} \quad (2.1)$$

$$R_n = \frac{a}{\sqrt{1-e^2 \sin^2 \varphi}} \quad (2.2)$$

其中,  $e = \frac{\sqrt{a^2 - b^2}}{a}$  为地球椭球偏心率;  $\varphi$  为目标点纬度, 单位为 rad。

基于面向对象的设计原则, 本次实验封装了地球椭球体模型。通过定义基础的地球椭球体类并实例化 GRS80 椭球体, 便于后续无缝切换或拓展其他地球模型。

表 2.2 地球椭球体类设计参照表

Ellipsoid 类	
private:	
double a = 0.;	长半轴
double b = 0.;	短半轴
double f = 0.;	椭球扁率
double e = 0.;	第一偏心率
double ee = 0.;	第二偏心率
double we = 0.;	地球自转角速度
double GM = 0.;	地球引力常数
void createWGS84();	创建 WGS84 椭球模型
void createCGCS2000();	创建 CGCS2000 椭球模型
void createGRS80();	创建 GRS80 椭球模型
public:	
Ellipsoid();	默认构造函数
Ellipsoid(const std::string &type);	根据传入的椭球类型构造不同椭球模型
double Rm(double latitude) const;	计算子午圈半径
double Rn(double latitude) const;	计算卯酉圈半径
double get_we() const;	获取地球自转角速度

### 2.1.2 正常重力模型

GRS80 地球椭球模型下正常重力计算公式如下

$$g_0 = 9.7803267715(1 + 0.0052790414 \sin^2 \varphi + 0.0000232718 \sin^4 \varphi) \quad (2.3)$$

$$g(\varphi, h) = g_0 - (3.087691089 \times 10^{-6} - 4.397731 \times 10^{-9} \sin^2 \varphi)h + 0.721 \times 10^{-12} h^2 \quad (2.4)$$

其中,  $g(\varphi, h)$  为纬度  $\varphi$  和高程  $h$  点处的正常重力, 单位  $m/s^2$ , 高程单位为  $m$ , 纬度单位为  $rad$ 。

考虑到该项正常重力计算公式系数仅针对 GRS80 地球椭球, 因此, 程序中单独设计了函数来实现正常重力计算。

表 2.3 GRS80 正常重力计算函数参照表

函数名	含义
double GRS80_G0(double latitude);	计算地球表面重力
double GRS80_G(double latitude, double h);	计算目标位置地球正常重力

### 2.1.3 地球自转角速度向量与位移角速度向量

在纯惯导动态解算中, 导航坐标系(n 系)并非静止不动的, 而是随着地球自转以及载体位移在空间中不断旋转。因此, 在更新解算中不仅需要扣除地球自转带来的基准偏差, 还要补偿因为载体运动导致的 n 系变化。由于地球自转角速度向量与位移角速度向量与地球椭球模型息息相关, 因此, 在本模块中介绍其原理。

地球自转角速度  $w_e$  在 NED 导航系下的投影取决于当地纬度  $\varphi$ :

$$w_{ie}^n = \begin{bmatrix} w_e \cos \varphi \\ 0 \\ -w_e \sin \varphi \end{bmatrix} \quad (2.5)$$

当载体在地球椭球表面运动时, 由于导航系始终贴合当地水平面, 运动本身会引起 n 系相对于地球系(e 系)的转动。其分量与载体速度与椭球曲率半径相关:

$$w_{en}^n = \begin{bmatrix} \frac{v_E}{(R_N + h)} \\ \frac{-v_N}{(R_M + h)} \\ \frac{-v_E \tan \varphi}{(R_N + h)} \end{bmatrix} \quad (2.6)$$

其中,  $v_i$  为  $i$  方向的速度, 单位为  $m/s$ ;  $R_N, R_M$  分别为目标点的卯酉圈半径和子午圈半径, 计算公式见(2.1)-(2.2); 其他变量同上, 单位均为国际标准单位。

由于地球自转角速度向量和位移角速度向量与实时解算的位置、速度有关, 需要频繁更新, 考虑到模块的独立性, 本程序不选择在地球常数模块计算该项, 而是选择在更新算法中实时计算该项补偿。

## 2.2 姿态表达式的相互转换

在捷联惯导解算中，为提高计算效率并确保结果直观，通常采取多种姿态表达式混合计算。因此，不同姿态表达式之间的相互转换至关重要。本模块中将直接给出 **b** 系相对于参考系 **R** 系的姿态表达式之间的转换公式(主要针对算法实际用到的姿态转换)。

规定：

**b** 系为前右下坐标系；

欧拉角按“ZYX”的旋转顺序定义为航向角  $yaw(\psi)$ 、俯仰角  $pitch(\theta)$  以及横滚角  $roll(\phi)$ ；

等效旋转矢量  $\phi_{rb}$  描述为 **R** 系绕空间矢量  $\phi_{rb}$  的方向转动角度  $\|\phi_{rb}\|$  后与 **b** 系对齐；

方向余弦矩阵  $C_b^R$  与姿态四元数  $q_b^R$  角标含义相同，描述为从 **b** 系旋转到 **R** 系。

### 2.2.1 欧拉角转方向余弦矩阵

$$C_b^R = \begin{bmatrix} c\theta c\psi & -c\phi s\psi + s\phi s\theta c\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & -s\phi c\psi + c\phi s\theta s\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \quad (2.7)$$

式中， $c = \cos()$ ， $s = \sin()$ ；

### 2.2.2 欧拉角转四元数

$$q_b^R = \begin{bmatrix} \cos \frac{\phi}{2} \cos \frac{\theta}{2} \cos \frac{\psi}{2} + \sin \frac{\phi}{2} \sin \frac{\theta}{2} \sin \frac{\psi}{2} \\ \sin \frac{\phi}{2} \cos \frac{\theta}{2} \cos \frac{\psi}{2} - \cos \frac{\phi}{2} \sin \frac{\theta}{2} \sin \frac{\psi}{2} \\ \cos \frac{\phi}{2} \sin \frac{\theta}{2} \cos \frac{\psi}{2} + \sin \frac{\phi}{2} \cos \frac{\theta}{2} \sin \frac{\psi}{2} \\ \cos \frac{\phi}{2} \cos \frac{\theta}{2} \sin \frac{\psi}{2} - \sin \frac{\phi}{2} \sin \frac{\theta}{2} \cos \frac{\psi}{2} \end{bmatrix} \quad (2.8)$$

### 2.2.3 方向余弦矩阵转欧拉角

方向余弦矩阵  $C_b^R$  第  $i$  行第  $j$  列记为  $c_{ij}$ ， $1 \leq i, j \leq 3$ 。首先计算俯仰角 ( $|\theta| \leq \frac{\pi}{2}$ ):

$$\theta = \tan^{-1} \left( \frac{-c_{31}}{\sqrt{c_{32}^2 + c_{33}^2}} \right) \quad (2.9)$$



当 $|\theta| \neq \frac{\pi}{2}$  (实际计算以 $|c_{31}| < 0.999$  判断), 横滚角和航向角( $-\pi \leq \phi, \psi \leq \pi$ ) 计算公式如下:

$$\begin{aligned}\phi &= \tan^{-1} \frac{c_{32}}{c_{33}} \\ \psi &= \tan^{-1} \frac{c_{21}}{c_{11}}\end{aligned}\quad (2.10)$$

当 $|c_{31}| \geq 0.999$  即出现万向锁的情况时, 只能计算航向角和横滚角的组合:

$$\begin{cases} \psi - \phi = \tan^{-1} \left( \frac{c_{23} - c_{12}}{c_{13} + c_{22}} \right), & c_{31} \leq -0.999, \\ \psi + \phi = \pi + \tan^{-1} \left( \frac{c_{23} + c_{12}}{c_{13} - c_{22}} \right), & c_{31} \geq 0.999. \end{cases}\quad (2.11)$$

#### 2.2.4 方向余弦矩阵转四元数

首先做如下计算:

$$\begin{aligned}P_1 &= 1 + \text{tr}(\mathbf{C}_b^R), P_2 = 1 + 2c_{11} - \text{tr}(\mathbf{C}_b^R), \\ P_3 &= 1 + 2c_{22} - \text{tr}(\mathbf{C}_b^R), P_4 = 1 + 2c_{33} - \text{tr}(\mathbf{C}_b^R)\end{aligned}\quad (2.12)$$

(1) 如果  $P_1 = \max(P_1, P_2, P_3, P_4)$ , 则

$$q_0 = 0.5\sqrt{P_1}, q_1 = \frac{c_{32} - c_{23}}{4q_0}, q_2 = \frac{c_{13} - c_{31}}{4q_0}, q_3 = \frac{c_{21} - c_{12}}{4q_0}\quad (2.13)$$

(2) 如果  $P_2 = \max(P_1, P_2, P_3, P_4)$ , 则

$$q_1 = 0.5\sqrt{P_2}, q_2 = \frac{c_{21} + c_{12}}{4q_1}, q_3 = \frac{c_{13} + c_{31}}{4q_1}, q_0 = \frac{c_{32} - c_{23}}{4q_1}\quad (2.14)$$

(3) 如果  $P_3 = \max(P_1, P_2, P_3, P_4)$ , 则

$$q_2 = 0.5\sqrt{P_3}, q_3 = \frac{c_{32} + c_{23}}{4q_2}, q_0 = \frac{c_{13} - c_{31}}{4q_2}, q_1 = \frac{c_{12} + c_{21}}{4q_2}\quad (2.15)$$

(4) 如果  $P_4 = \max(P_1, P_2, P_3, P_4)$ , 则

$$q_3 = 0.5\sqrt{P_4}, q_0 = \frac{c_{21} - c_{12}}{4q_3}, q_1 = \frac{c_{13} + c_{31}}{4q_3}, q_2 = \frac{c_{32} + c_{23}}{4q_3}\quad (2.16)$$

#### 2.2.5 四元数转方向余弦矩阵

$$\mathbf{C}_b^R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}\quad (2.17)$$

#### 2.2.6 四元数转等效旋转矢量

(1) 如果  $q_0 \neq 0$ , 则

$$\|0.5\phi_{Rb}\| = \tan^{-1} \left( \frac{\sin\|0.5\phi_{Rb}\|}{\cos\|0.5\phi_{Rb}\|} \right) = \tan^{-1} \left( \frac{\sqrt{q_1^2 + q_2^2 + q_3^2}}{q_0} \right) \quad (2.18)$$

$$f = \frac{\sin\|0.5\phi_{Rb}\|}{\|\phi_{Rb}\|} \quad (2.19)$$

$$\phi_{Rb} = \frac{1}{f} [q_1 \quad q_2 \quad q_3]^\top \quad (2.20)$$

(2) 如果  $q_0 = 0$ ，则

$$\phi_{Rb} = \pi [q_1 \quad q_2 \quad q_3]^\top \quad (2.21)$$

## 2.2.7 等效旋转矢量转方向余弦矩阵

$$C_b^R = I + \frac{\sin\|\phi_{Rb}\|}{\|\phi_{Rb}\|} (\phi_{Rb} \times) + \frac{1 - \cos\|\phi_{Rb}\|}{\|\phi_{Rb}\|^2} (\phi_{Rb} \times)(\phi_{Rb} \times) \quad (2.22)$$

$$(\phi_{Rb} \times) = \begin{bmatrix} 0 & -\phi_{Rbz} & \phi_{Rby} \\ \phi_{Rbz} & 0 & -\phi_{Rbx} \\ -\phi_{Rby} & \phi_{Rbx} & 0 \end{bmatrix} \quad (2.23)$$

## 2.2.8 等效旋转矢量转四元数

$$\mathbf{q}_b^R = \begin{bmatrix} \cos\|0.5\phi_{Rb}\| \\ \frac{\sin\|0.5\phi_{Rb}\|}{\|0.5\phi_{Rb}\|} 0.5\phi_{Rb} \end{bmatrix} \quad (2.24)$$

为集中管理多种姿态描述并实现其相互转换，本程序封装了一个姿态类，用以存储各类姿态数据并提供高效的转换方式。由于转换逻辑较简单，仅需要注意分母为零时的容错处理(泰勒展开)，因此，这里不对转换函数实现细节展开说明，仅从整体上给出姿态类的设计参照表。

表 2.4 姿态类设计参照表

Attitude 类	
public:	
Eigen::Matrix3d Cb_n;	方向余弦矩阵(b 系到 n 系)
double yaw, pitch, roll;	载体姿态角(航向、俯仰、横滚) ZYX
Eigen::Quaterniond Qb_n;	姿态四元数 (b 系到 n 系)
Eigen::Vector3d EqualRotateVec;	等效旋转矢量(n:b)
void Euler2DCM();	欧拉角转方向余弦矩阵

---

<code>void Euler2Quaternion();</code>	欧拉角转四元数
<code>void DCM2Euler();</code>	方向余弦转欧拉角
<code>void DCM2Quaternion();</code>	方向余弦转四元数
<code>void Quaternion2ERV();</code>	四元数转等效旋转矢量
<code>void Quaternion2DCM();</code>	四元数转方向余弦
<code>void ERV2DCM();</code>	等效旋转矢量转方向余弦
<code>void ERV2Quaternion();</code>	等效旋转矢量转四元数

---

## 2.3 惯导机械编排算法

捷联惯导机械编排在不同参考系下存在不同的数学处理方案，但它们处理数据的整体流程大同小异。为了让解算结果更加直观对应我们熟知的经纬高，本实验重点围绕  $n$  系下的机械编排算法展开说明。

### 2.3.1 速度更新算法

速度更新的核心公式如下：

$$\mathbf{v}_k^n = \mathbf{v}_{k-1}^n + \Delta \mathbf{v}_{f,k}^n + \Delta \mathbf{v}_{g/cor,k}^n \quad (2.25)$$

其中， $\mathbf{v}_k^n$  是  $k$  时刻 IMU 相对于  $n$  系的速度， $\mathbf{v}_{k-1}^n$  为  $k-1$  时刻 IMU 相对于  $n$  系的速度； $\Delta \mathbf{v}_{g/cor,k}^n$  是与有害加速度相关的速度变化量，称作重力哥式积分项； $\Delta \mathbf{v}_{f,k}^n$  表现与比力测量值相关的速度变化量，称作比力积分项。

对于重力哥式积分项，

$$\Delta \mathbf{v}_{g/cor,k}^n = \int_{t_{k-1}}^{t_k} [\mathbf{g}_p^n(t) - (2\mathbf{w}_{ie}^n(t) + \mathbf{w}_{en}^n(t)) \times \mathbf{v}^n(t)] dt \quad (2.26)$$

一般而言， $\mathbf{g}_p^n(t)$  和  $\mathbf{w}_{en}^n(t)$  随位置和速度变化都是相对平缓连续，在 IMU 高采样率(100Hz...)的基础上，该项引起的变化非常小。因此，通常可将被积函数近似为随时间线性变化，即重力哥式积分项可简化为梯形积分

$$\begin{aligned} \Delta \mathbf{v}_{g/cor,k}^n &= \frac{1}{2} (\mathbf{a}_{gc}^n(t_{k-1}) + \mathbf{a}_{gc}^n(t_k)) \Delta t = \mathbf{a}_{gc}^n(t_{(k-1)/2}) \Delta t \\ \mathbf{a}_{gc}^n(t) &= \mathbf{g}_p^n(t) - (2\mathbf{w}_{ie}^n(t) + \mathbf{w}_{en}^n(t)) \times \mathbf{v}^n(t) \end{aligned} \quad (2.27)$$

$\mathbf{w}_{ie}^n$  和  $\mathbf{w}_{en}^n$  计算公式可参考公式(2.5)~(2.6)，但是由于此时尚未求得  $t_k$  时刻的位置速度，所以一般采取外推的形式求得中间时刻的位置速度，进而计算重力哥式积

分项。中间时刻位置速度可以通过简单的线性外推得到：

$$x_{(k-1)/2} = x_{k-1} + \frac{1}{2}(x_{k-1} - x_{k-2}) = \frac{3}{2}x_{k-1} - \frac{1}{2}x_{k-2}; (x = r, v) \quad (2.28)$$

对于比力积分项，

$$\Delta v_{f,k}^n = \int_{t_{k-1}}^{t_k} C_b^n(t) f^b(t) dt \quad (2.29)$$

由于在积分时段内载体的姿态以及比力向量都可能随时间快速变化，因此，无法像哥式积分项那样近似处理。一般而言，视积分时段内  $n$  系相对于惯性系的转动为时间的线性函数并且积分区间内  $n$  的转动为小量，可将上式近似为

$$\Delta v_{f,k}^n = \left[ I - \frac{1}{2}(\zeta_{n(k-1)n(k)} \times) \right] C_{b(k-1)}^n \Delta v_{f,k}^{b(k-1)} \quad (2.30)$$

其中， $\zeta_{n(k-1)n(k)} \approx [w_{ie}^n(t_{(k-1)/2}) + w_{en}^n(t_{(k-1)/2})] \Delta t$ ，对于中间时刻的处理与重力哥式积分项相同； $\Delta v_{f,k}^{b(k-1)} = \int_{t_{k-1}}^{t_k} C_{b(t)}^{b(k-1)} f^b(t) dt$ ，称作  $b$  系比力积分项。

对于  $b$  系比力积分项，本次实验中采取双子样假设，可以推导得到

$$\Delta v_{f,k}^{b(k-1)} = \Delta v_k + \frac{1}{2} \Delta \theta_k \times \Delta v_k + \frac{1}{12} (\Delta \theta_{k-1} \times \Delta v_k + \Delta v_{k-1} \times \Delta \theta_k) \quad (2.31)$$

其中， $\Delta v_k$ ， $\Delta \theta_k$  分别为  $k$  时刻加速度计增量观测 (m/s) 和陀螺增量观测(rad)，其他以此类推。

由上述公式可以看出，捷联惯导速度更新的关键步骤可归结为重力哥式积分项与比力积分项的补偿，程序中针对这两项进行计算后即可线性推算得到当前时刻的速度，具体实现流程如下图：

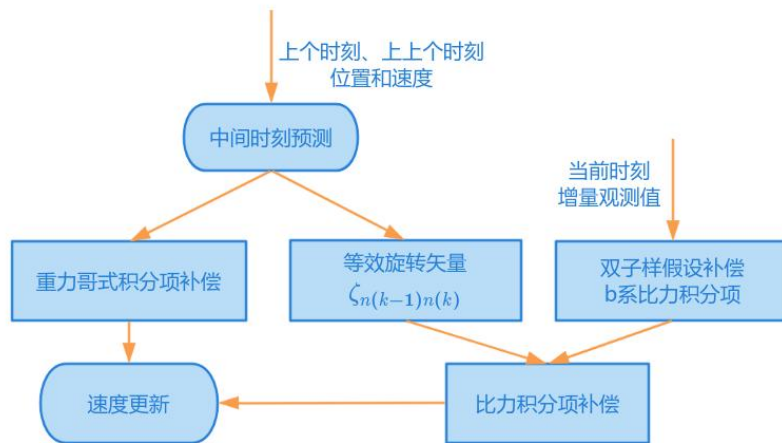


图 2.1 速度更新设计流程参照图

按照上述设计流程，依次实现相应的逻辑，其对应实现细节如下：

中间时刻预测:

```
01 // 预测中间时刻的参数
02 PVA pva_mid;
03 pva_mid.blh = 1.5 * pva_pre2[0].blh - 0.5 * pva_pre2[1].blh;
04 pva_mid.vel_n = 1.5 * pva_pre2[0].vel_n - 0.5 * pva_pre2[1].vel_n;
05
06 // 获取中间时刻的地球参数(GRS80)
07 double gp_mid = GRS80_G(pva_mid.blh[0], pva_mid.blh[2]);
08 Eigen::Vector3d gpn_mid;
09 gpn_mid << 0., 0., gp_mid;
10 double Rm_mid = GRS80.Rm(pva_mid.blh[0]), Rn_mid = GRS80.Rn(pva_mid.blh[0]);
11 double we_mid = GRS80.get_we();
12
13 // 中间时刻的地球自转角速度和位移角速度
14 Eigen::Vector3d wie_mid, wen_mid;
15 wie_mid << we_mid * cos(pva_mid.blh[0]), 0.0, -we_mid * sin(pva_mid.blh[0]);
16 wen_mid << pva_mid.vel_n[1] / (Rn_mid + pva_mid.blh[2]),
17         -pva_mid.vel_n[0] / (Rm_mid + pva_mid.blh[2]),
18         -pva_mid.vel_n[1] * tan(pva_mid.blh[0]) / (Rn_mid + pva_mid.blh[2]);
```

重力哥式项补偿:

```
01 // 重力哥式积分项
02 dv_gc = (gpn_mid - (2 * wie_mid + wen_mid).cross(pva_mid.vel_n)) * imu_cur.dt;
```

等效旋转矢量  $\zeta_{n(k-1)n(k)}$ :

```
01 // 等效旋转矢量 n(k-1)n(k)(线性外推)
02 Attitude att_nk1_nk;
03 att_nk1_nk.EqualRotateVec = (wie_mid + wen_mid) * imu_cur.dt;
04 Eigen::Matrix3d ssERM;
05 ssERM << 0, -att_nk1_nk.EqualRotateVec(2), att_nk1_nk.EqualRotateVec(1),
06         att_nk1_nk.EqualRotateVec(2), 0, -att_nk1_nk.EqualRotateVec(0),
07         -att_nk1_nk.EqualRotateVec(1), att_nk1_nk.EqualRotateVec(0), 0;
```

b 系比力积分项:

```
01 // b 系比力积分项(双子样假设)
02 Eigen::Vector3d dv_fb = imu_cur.dvel + 0.5 * imu_cur.dangel.cross(imu_cur.dvel)
03 + (1.0 / 12.0) * (imu_pre.dangel.cross(imu_cur.dvel) + imu_pre.dvel.cross(i
04 mu_cur.dangel));
```

比力积分项补偿:

```
01 // 比力积分项
02 dv_fn = (Eigen::Matrix3d::Identity() - 0.5 * ssERM) * pva_pre2[0].att.Cb_n * d
03 v_fb;
```

速度更新:

```
01 // 3.速度更新
02 pva_cur.vel_n = pva_pre2[0].vel_n + dv_gc + dv_fn;
```

### 2.3.2 位置更新算法

对比速度更新算法, 采样间隔内位置更新的计算误差一般相对较小, 因此, 可采用比较简单的梯形积分进行简化处理。在惯性导航中载体的位置一般采取大地坐标(即经纬高)表示。

因此, 可以首先进行高程更新

$$h_k = h_{k-1} - \frac{1}{2}(v_{D,k-1} + v_{D,k})\Delta t \quad (2.32)$$

其中,  $h_k$  为  $k$  时刻的高程;  $v_{D,k}$  为  $k$  时刻的垂向速度;  $\Delta t = t_k - t_{k-1}$  表示采样间隔; 其他变量以此类推。

然后进行高程纬度更新

$$\begin{aligned} \varphi_k &= \varphi_{k-1} + \frac{v_{N,k} + v_{N,k-1}}{2(R_{M,k-1} + \bar{h})}\Delta t \\ \bar{h} &= \frac{1}{2}(h_k + h_{k-1}) \end{aligned} \quad (2.33)$$

其中,  $\varphi_k$  表示  $k$  时刻的纬度;  $R_{M,k-1}$  为  $k-1$  时刻的子午圈半径;  $v_{N,k}$  为  $k$  时刻的北向速度; 其他变量以此类推。

最后可以进行经度更新

$$\begin{aligned} \lambda_k &= \lambda_{k-1} + \frac{v_{E,k} + v_{E,k-1}}{2(R_{N,(k-1)/2} + \bar{h})\cos \bar{\varphi}}\Delta t \\ \bar{\varphi} &= \frac{1}{2}(\varphi_k + \varphi_{k-1}) \end{aligned} \quad (2.34)$$

其中,  $\lambda_k$  表示  $k$  时刻的经度;  $R_{N,(k-1)/2}$  为中间时刻的卯酉圈半径;  $v_{E,k}$  为  $k$  时刻的东向速度; 其他变量以此类推。

由于该部分实现公式较为简单, 程序设计中按照顺序依次计算高程、纬度、经度即可, 因此, 这里直接给出具体实现:

位置更新(blh,deg,deg,m):

```

01 // 1.高程更新
02 pva_cur.blh[2] = pva_pre.blh[2] - 0.5 * (pva_pre.vel_n[2] + pva_cur.vel_n[2])
03 * imu_cur.dt;
04
05 // 2.纬度更新
06 double h_mean = 0.5 * (pva_pre.blh[2] + pva_cur.blh[2]);
07 double Rm = GRS80.Rm(pva_pre.blh[0]);
08 pva_cur.blh[0] = pva_pre.blh[0] + (pva_pre.vel_n[0] + pva_cur.vel_n[0]) * imu_
09 cur.dt / (2 * (Rm + h_mean));
10
11 // 3.经度更新
12 double latitude_mean = 0.5 * (pva_pre.blh[0] + pva_cur.blh[0]);
13 double Rn = GRS80.Rn(latitude_mean);
14 pva_cur.blh[1] = pva_pre.blh[1] + (pva_pre.vel_n[1] + pva_cur.vel_n[1]) * imu_
15 cur.dt / (2 * (Rn + h_mean) * cos(latitude_mean));

```

### 2.3.3 姿态更新

以姿态四元数表示的姿态更新的核心公式如下：

$$q_{b(k)}^{n(k)} = q_{n(k-1)}^{n(k)} \circ q_{b(k-1)}^{n(k-1)} \circ q_{b(k)}^{b(k-1)} \quad (2.35)$$

其中， $q_{b(k)}^{b(k-1)}$  表示 b 系姿态变换对应的四元数

$$q_{b(k)}^{b(k-1)} = \begin{bmatrix} \cos \|0.5\phi_k\| \\ \frac{\sin \|0.5\phi_k\|}{\|0.5\phi_k\|} 0.5\phi_k \end{bmatrix} \quad (2.36)$$

其中， $\phi_k = \phi_{b(k-1)b(k)}$  表示 b 系相对 i 系转动的等效旋转矢量，可以通过双子样假设求得

$$\phi_k = \Delta\theta_k + \frac{1}{12}\Delta\theta_{k-1} \times \Delta\theta_k \quad (2.37)$$

$q_{n(k-1)}^{n(k)}$  表示 n 系相对 i 系姿态变化对应的姿态四元数

$$q_{n(k-1)}^{n(k)} = \begin{bmatrix} \cos \|0.5\zeta_k\| \\ -\frac{\sin \|0.5\zeta_k\|}{\|0.5\zeta_k\|} 0.5\zeta_k \end{bmatrix} \quad (2.38)$$

其中， $\zeta_k = \zeta_{n(k-1)n(k)}$  为 n 系相对 i 系转动的等效旋转矢量，可以通过积分时段内地球自转角速度和转动角速度的梯形积分近似得到

$$\zeta_k \approx \left[ w_{ie}^n(t_{(k-1)/2}) + w_{en}^n(t_{(k-1)/2}) \right] \Delta t \quad (2.39)$$

从上述公式可以看出，和速度更新一样，姿态更新主要分为三个四元数相乘：

$q_{b(k-1)}^{n(k-1)}$  为上个历元的姿态四元数，为已知数； $q_{b(k)}^{b(k-1)}$  为积分时段内  $b$  系的变化，可以通过 IMU 测量值计算得到； $q_{n(k-1)}^{n(k)}$  为积分时段内  $n$  系的变化，可以通过中间时刻地球参数计算得到，并且由于此时当前时刻的位置速度已经得到，可以采取内插而不是外推的方法更精确得到中间时刻的参数。因此，姿态更新的完整程序设计参照图和具体实现如下：

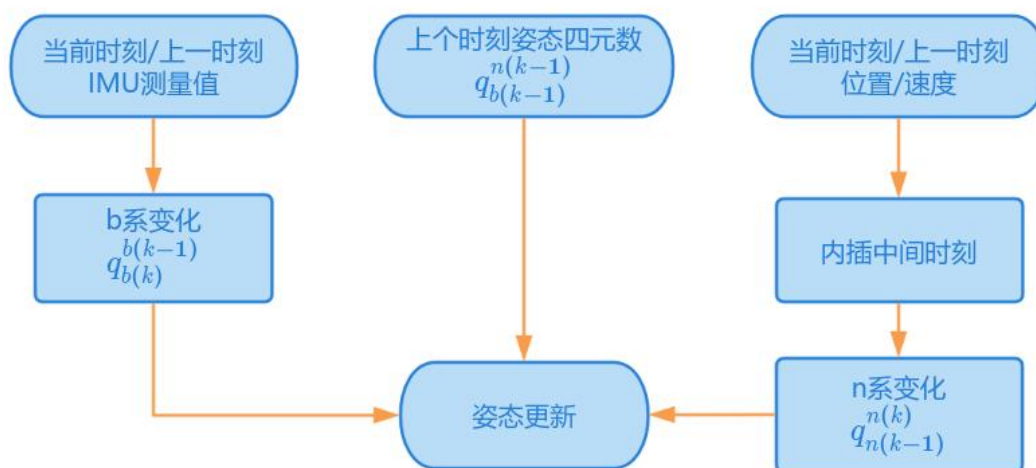


图 2.2 姿态更新设计流程参照图

$b$  系变化  $q_{b(k)}^{b(k-1)}$ ：

```

01 // 1.b 系相当于 i 系的等效旋转矢量(b(k-1):b(k))
02 Abk1_k.EqualRotateVec = imu_cur.dangel + (1.0 / 12.0) * imu_pre.dangel.cross(i
03 mu_cur.dangel);
04 Abk1_k.ERV2Quaternion();
  
```

中间时刻内插：

```

01 // 中间时刻的位置和速度
02 PVA pva_mid;
03 pva_mid.blh = (pva_cur.blh + pva_pre.blh) / 2.0;
04 pva_mid.vel_n = (pva_cur.vel_n + pva_pre.vel_n) / 2.0;
05
06 // 获取中间时刻的地球参数(GRS80)
07 double gp_mid = GRS80_G(pva_mid.blh[0], pva_mid.blh[2]);
08 Eigen::Vector3d gpn_mid;
09 gpn_mid << 0., 0., gp_mid;
10 double Rm_mid = GRS80.Rm(pva_mid.blh[0]), Rn_mid = GRS80.Rn(pva_mid.blh[0]);
11 double we_mid = GRS80.get_we();
12
13 // 中间时刻的地球自转角速度和位移角速度
14 Eigen::Vector3d wie_mid, wen_mid;
  
```



```

15 wie_mid << we_mid * cos(pva_mid.blh[0]), 0.0, -we_mid * sin(pva_mid.blh[0]);
16 wen_mid << pva_mid.vel_n[1] / (Rn_mid + pva_mid.blh[2]),
17     -pva_mid.vel_n[0] / (Rm_mid + pva_mid.blh[2]),
18     -pva_mid.vel_n[1] * tan(pva_mid.blh[0]) / (Rn_mid + pva_mid.blh[2]);
    
```

n 系变化  $q_{n(k-1)}^{n(k)}$  :

```

01 Ank1_k.EqualRotateVec = -(wie_mid + wen_mid) * imu_cur.dt;
02 Ank1_k.ERV2Quarternion();
    
```

姿态更新:

```

01 pva_cur.att.Qb_n = Ank1_k.Qb_n * pva_pre.att.Qb_n * Abk1_k.Qb_n;
02 pva_cur.att.Qb_n.normalize();
03 pva_cur.att.Quarternion2DCM();
04 pva_cur.att.DCM2Euler();
    
```

由于更新过程中需要频繁用到等效旋转矩阵和姿态四元数之间的转换，程序中统一使用定义的  $q_b^R$  与  $\phi_{Rb}$  形式进行转换。

## 2.4 零速判断原理

静止状态下 IMU 输出的理论值为当地重力与地球自转角速度。依据这一特性，可设计一种简易的静止判断方法，即

$$\begin{cases} |w_{ib}^b - w_{ie}^n| \leq \delta w, |f^b - g_p^n| \leq \delta f^b & \text{static} \\ |w_{ib}^b - w_{ie}^n| > \delta w, |f^b - g_p^n| > \delta f^b & \text{else} \end{cases} \quad (2.40)$$

其中， $w_{ib}^b$  与  $f^b$  分别为测量角速度与比力； $w_{ie}^n$  与  $g_p^n$  分别为地球自转角速度与正常重力； $\delta w$  与  $\delta f^b$  分别为角速度与比力零速判断的阈值。

由于该判断逻辑较为简单，这里直接给出程序设计过程中的具体实现

零速判断

```

01 double acc_norm = imu_cur.dvel.norm() / imu_cur.dt;
02 double gyro_norm = imu_cur.dangel.norm() / imu_cur.dt;
03 bool is_acc_static = std::abs(acc_norm - 9.806) < STATIC_THRESHOLD_ACCEL;
04 bool is_gyro_static = gyro_norm < STATIC_THRESHOLD_GYRO;
05 return is_acc_static && is_gyro_static;
    
```

遵循第1章所述的整体实验流程，本实验可分为外业数据采集与内业数据处理两阶段。外业阶段主要涵盖所有硬件相关的现场操作，以获取原始观测数据；内业阶段则旨在通过算法处理，从原始数据中提取有效的导航信息。两个阶段通过数据文件的读写进行衔接。本章将分别对这两个阶段的具体要求进行详细说明。

### 3.1 外业数据采集

#### 3.1.1 实验设备

实验中采用华测 CGI-1230 型号惯性导航系统(INS)，GNSS 天线、GNSS 基站接收机以及实验小推车等设备，实验地点选定在武汉大学友谊广场附近区域。



图 3.1 实验设备图

#### 3.1.2 数据采集流程

- (1) 设备组装与杆臂测量：完成各仪器设备之间的连接，并精确测定 GNSS 天线相位中心相对于 IMU 中心在  $b$  系下的空间向量。
- (2) 上电与轴线确认：检查无误后，仪器设备上电，并通过对小车做一些验证动作以事后观察波形进行判断和核实 IMU 的敏感轴线。
- (3) 初始静止：将小车推至预设起点，静置 5 分钟用于初始对准。
- (4) 动态测试：小组成员轮流推行，按预设的“斜 8 字”轨迹运行。每人推行两圈后静止 1 分钟，为程序提供 ZUPT 修正机会。
- (5) 最终静止：小组成员都完成实验后将小车推回起点，重新静止 5 分钟用以获

取更高精度的参考真值。

(6) 拷贝与检查原始数据，确认无误后关闭电源。

### 3.1.3 实验信息说明

本组实验采集数据时的详细参数配置如下：

表 3.1 实验参数表

信息	信息说明
采集时段	11 月 3 日 11:40~12:30
惯导类型	华测 CGI-1230
IMU 轴系定义	右前上
杆臂向量(m)	0.162, 0.195, 0.269
数据转换软件	Converter3.0.7.exe
参考真值	GNSS/INS 组合导航结果

实验中具体的运行轨迹如下：

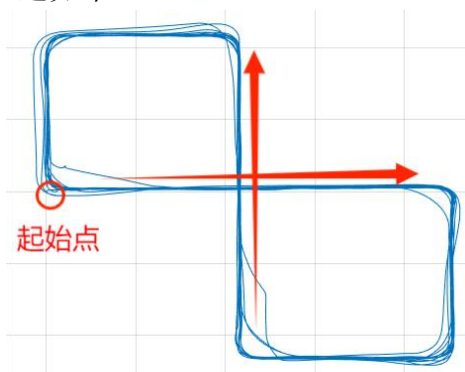


图 3.2 实际运行轨迹

实验初始静止对准图片如下：

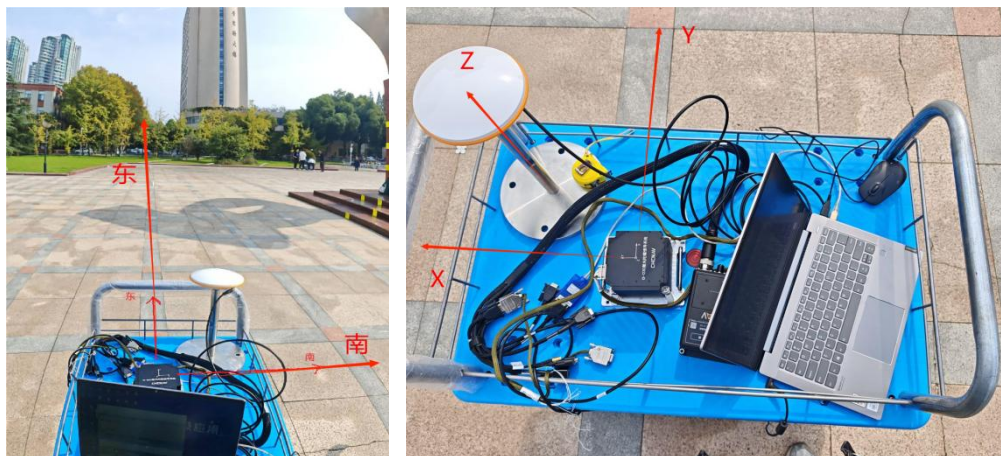


图 3.3 初始静态对准图

### 3.2 内业数据处理

在第2章中，我们重点阐述了捷联惯导解算中核心的数学原理及其离散化实现方法。因此，接下来的小节将更侧重于程序整体逻辑的衔接以及软件架构的构建。本次实验中纯惯导解算程序整体流程图如下：

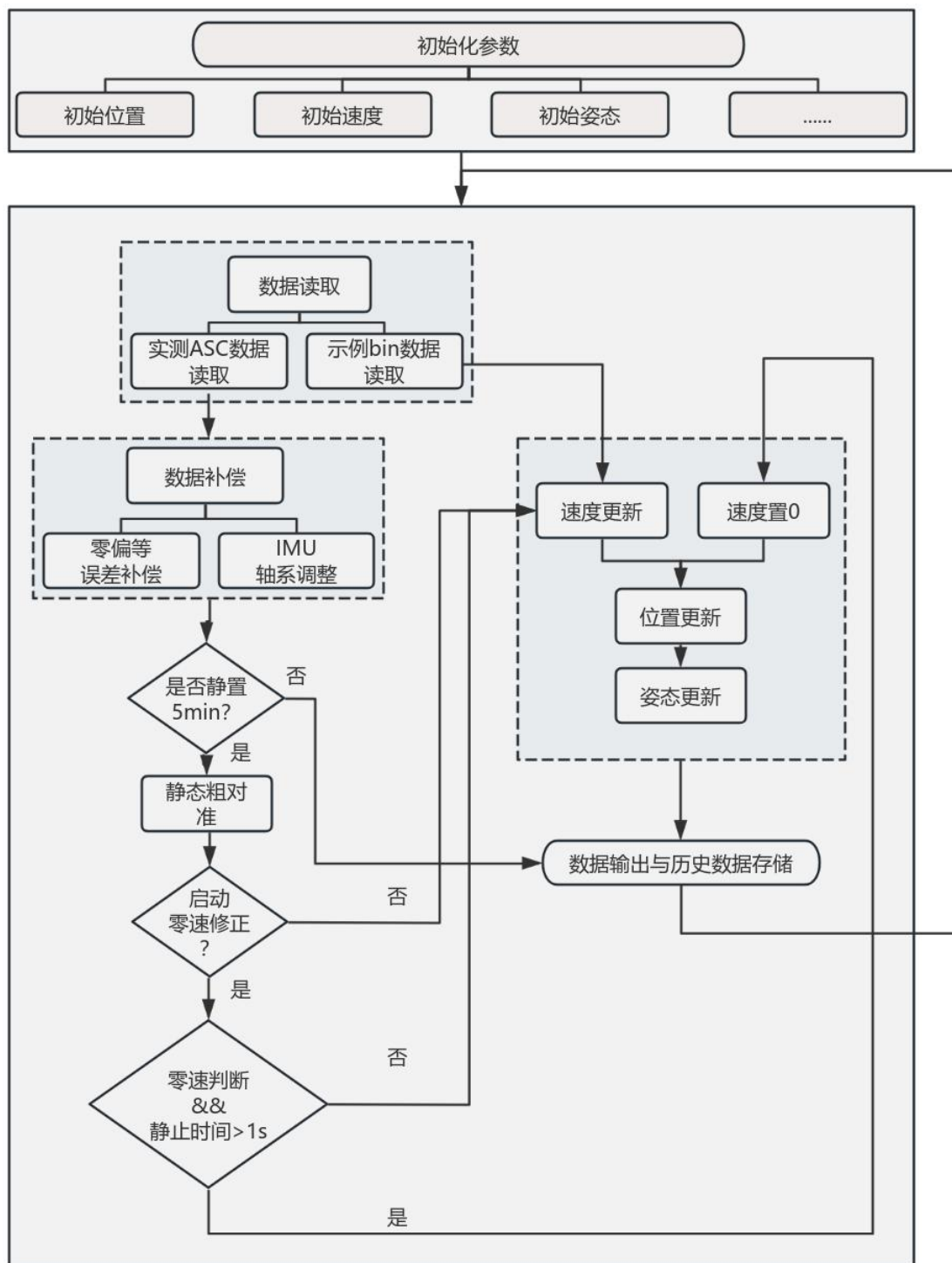


图 3.4 纯惯导全 demo 解算流程图

针对解算结果绘图分析部分，本实验采取 matlab 程序单独读取解算结果文件绘图，因此，流程图中并未将其归算到软件框架中。

### 3.2.1 初始化参数

参数初始化部分主要完成参数初始状态尤其是位置、速度以及姿态的设置，为后续算法提供准确的起算点。本实验中，针对示例数据，直接从参考真值中获取初始值进行硬编码；对于实测数据，位置速度根据参考真值进行硬编码，姿态角根据自编静态粗对准函数得到。具体信息如下：

表 3.2 初始参数配置表

数据来源	数据详情
示例数据	起算时间(s): 91620.00
	初始纬度(deg): 23.1373950708
	初始经度(deg): 113.3713651222
	初始高程(m): 2.175
	初始速度(m/s): [0, 0, 0]
	初始横滚角(deg): 0.0107951084511778
	初始俯仰角(deg): -2.14251290749072
实测数据	初始航向角(deg): -75.7498049314083
	起算时间(s): 100775.00
	粗对准时间(s): 300
	初始纬度(deg): 30.527907149
	初始经度(deg): 114.3558131205
	初始高程(m): 19.621
	初始速度(m/s): [0, 0, 0]

### 3.2.2 数据读取

由于示例数据采取原始二进制(bin)格式存储，而实测数据通过软件转换成了文本(ASC)格式存储，二者在数据结构与编码方式上存在显著差异。因此，在实验过程中，必须针对不同格式的数据采取相应的解析方法。

对于示例数据，其 IMU 数据格式如下



表 3.3 示例数据 IMU 原始数据格式(前右下)

序号	数据	单位	类型	字节数	说明
1	time	s	double	8	时间
2	gyro[0]	rad	double	8	X 轴陀螺数据(角增量)
3	gyro[1]	rad	double	8	Y 轴陀螺数据(角增量)
4	gyro[2]	rad	double	8	Z 轴陀螺数据(角增量)
5	accl[0]	rad	double	8	X 轴加速度计数据(速度增量)
6	accl[1]	rad	double	8	X 轴加速度计数据(速度增量)
7	accl[2]	rad	double	8	X 轴加速度计数据(速度增量)

按照上述格式，数据读取时按照指定字节指定变量顺序读取即可。

对于实测数据，其具体格式参考对应型号惯导接口文档，本次实验所采取的惯导参数如下：

表 3.4 实测数据惯导信息

惯导信息	信息说明
型号	华测 CGI-1230
设备加速度计比例系数(增量对应)	1/655360
设备陀螺仪比例系数(增量对应)	1/160849.543863

### 3.2.3 数据补偿

一般来讲，由于惯性测量仪器存在零偏、比例因子等误差，这些误差会在导航解算过程中不断累积，从而导致位置、速度和姿态的估计结果严重偏离真实值。因此，一般组合导航程序会利用 Kalman 滤波估计出这些误差予以补偿，而纯惯导通常可以通过实验室标定得到大致数据予以改正，但是由于本次实验中采取的华测 CGI-1230 惯导并没有参数标定，因此，这里并没有对实测数据进行误差改正。

由于实验中所用的 IMU 载体坐标系为右前上方向，而程序编排采取的坐标系为前右下坐标系，因此，数据补偿模块中还需要进行测量数据的坐标转换：

$$\begin{aligned}
 f_{a,x} &= f_{raw,y}, f_{a,y} = f_{raw,x}, f_{a,z} = -f_{raw,z} \\
 w_{a,x} &= w_{raw,y}, w_{a,y} = w_{raw,x}, w_{a,z} = -w_{raw,z}
 \end{aligned}
 \tag{2.41}$$

其中， $f_{a,i}$  与  $f_{raw,i}$  分别为改正后和改正前的比力观测值，对于角速度观测值  $w$  同理；按照增量形式的 IMU 观测值作同样的处理。

### 3.2.4 纯惯导解算

对于示例数据解算以及实测数据首次解算，本实验均采用无零速修正惯导解算，按照流程图逻辑解算即可。

对于带零速修正的惯导解算，程序中先通过零速判断函数判断当前历元是否静止，然后判断滑动窗口历元内(本程序采取 3 秒)是否均静止来确定是否进行零速更新，以防测量噪声造成的误判现象。实现伪代码如下：

---

零速更新伪代码

---

```
static_time = 0.0
FOR each epoch:
    imu_data = read_data()
    imu_data = compensate_data()
    is_static = detect_zero_velocity(imu_data)
    IF is_static:
        static_time +=  $\Delta t$ 
        IF static_time > T_threshold:
            update_with_zupt()
        ELSE:
            update_normal()
    ELSE:
        static_time = 0.0
        update_normal()
output_results()
```

---

### 3.2.5 其他说明

上文重点阐述了程序实现过程中各核心模块的设计要点与关键考量。限于篇幅，对于原理部分已经介绍的模块以及其他辅助性模块的具体实现细节，本文不再展开详述。程序具体实现见提供的源码注释，实现逻辑参考上述流程图。

## 武汉大学测绘学院《惯性导航原理》实验报告 第4章 结果和分析

本实验首先对原始解算结果进行对比与误差分析,计算其均值与均方根误差(RMSE)以量化性能;随后,将经纬高坐标转换至站心坐标系(ENU),绘制平面轨迹图,直观评估定位误差的分布与累积情况。通常,将大地坐标转换为站心系相对坐标可采用如下近似方法:

$$\begin{aligned} N &= (\varphi - \varphi_0)(R_M + h) \\ E &= (\lambda - \lambda_0)(R_N + h) \cos \varphi \end{aligned} \quad (2.42)$$

其中,  $\varphi$ ,  $\lambda$  表示任意时刻的纬度和经度(单位 rad),  $\varphi_0$ ,  $\lambda_0$  表示站心坐标系原点的纬度和经度(单位 rad), 一般可选轨迹起点为站心坐标系原点;  $N$ ,  $E$  为北坐标和东坐标(单位 m),  $h$  为椭球高(单位 m);  $R_M$ ,  $R_N$  为子午圈和卯酉圈半径, 计算公式见第2章数学原理。

### 4.1 示例数据结果分析

对比示例数据解算结果与参考结果的原始对比图(如下), 可以看出解算结果与参考结果曲线变化趋势与幅度基本一致, 说明自编程序并不存在显著的大错误。

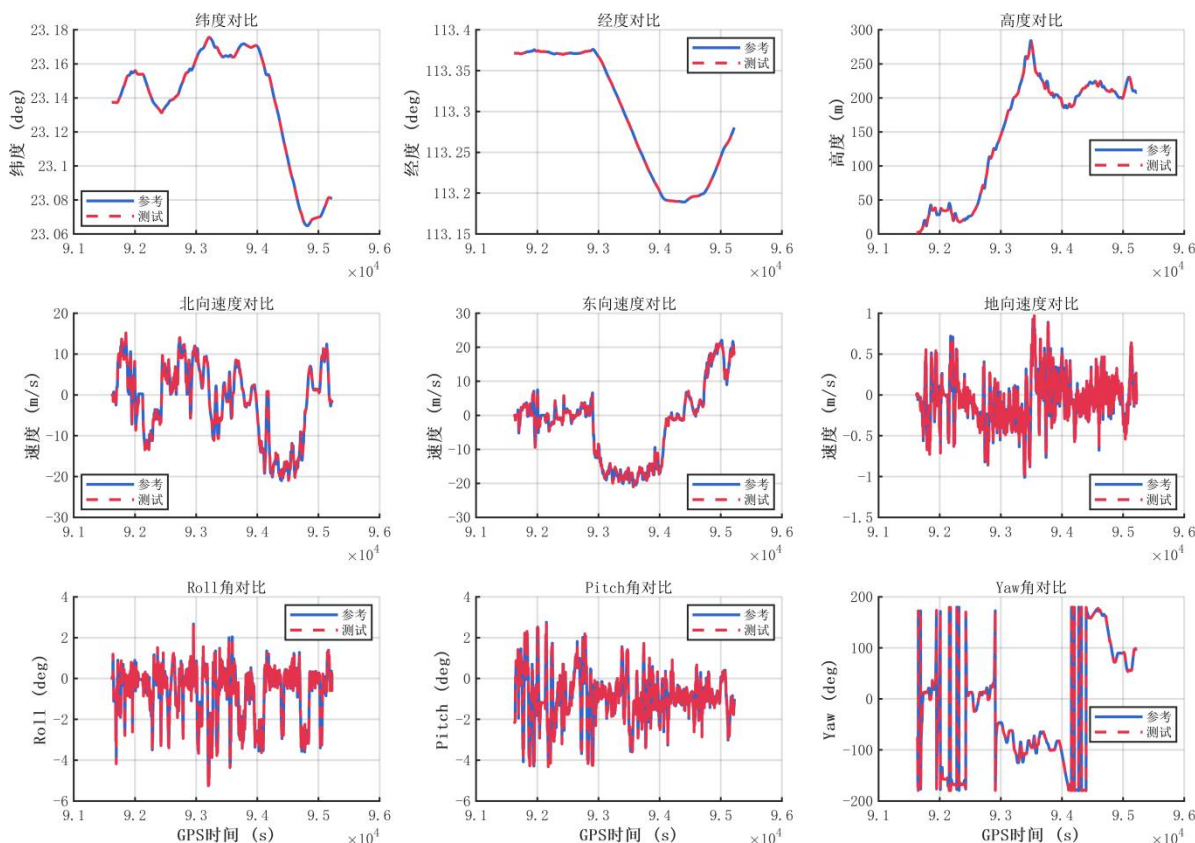


图 4.1 示例数据原始结果对比图



下面直接给出解算结果与参考结果之间的差分误差图以及统计指标：

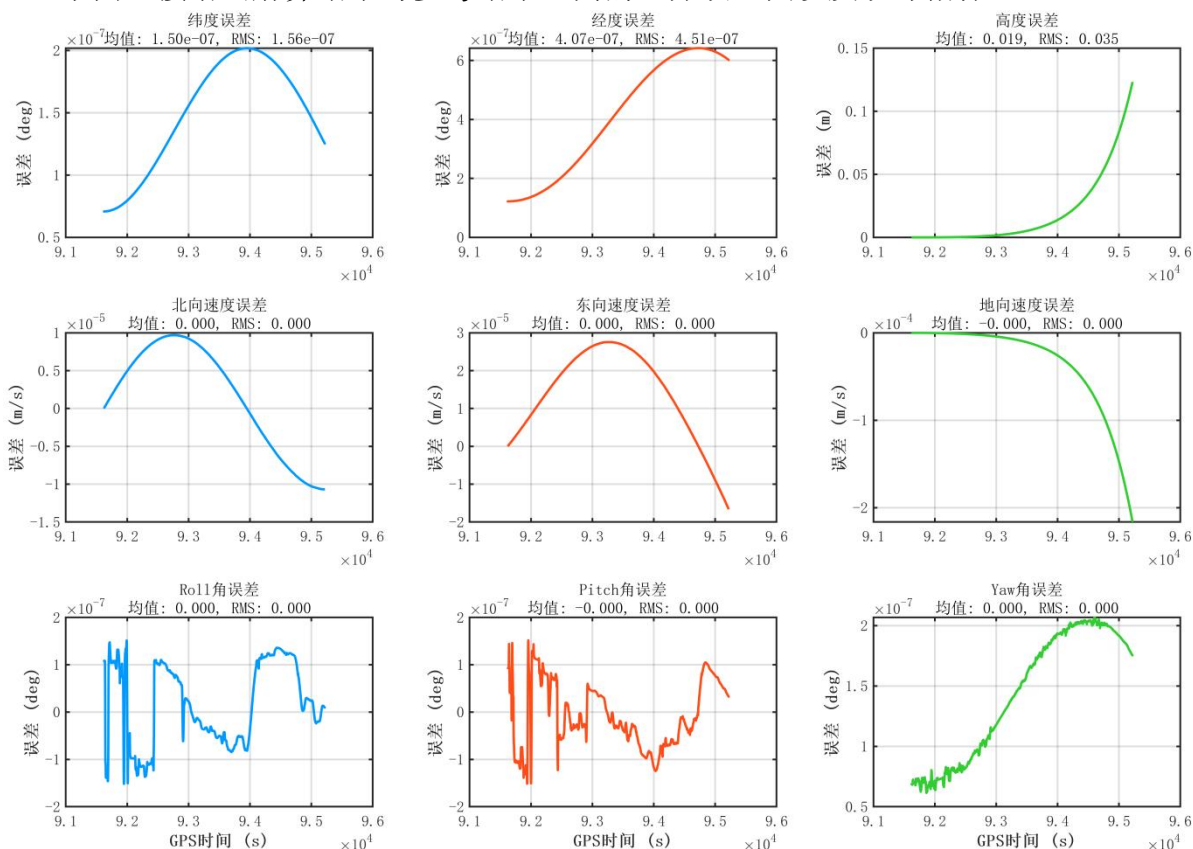


图 4.2 示例数据误差图

表 4.1 示例数据误差统计表

参数	均值	RMS	最大值	最小值
纬度(deg)	0.00000015	0.00000016	0.00000020	0.00000007
经度(deg)	0.00000041	0.00000045	0.00000064	0.00000012
高度(m)	0.01890034	0.03450490	0.12316692	0.00000000
北向速度(m/s)	0.00000162	0.00000713	0.00000969	-0.00001069
东向速度(m/s)	0.00001371	0.00001820	0.00002756	-0.00001665
地向速度(m/s)	-0.00003420	0.00006105	0.00000001	-0.00021630
横滚 roll(deg)	0.00000002	0.00000009	0.00000015	-0.00000015
俯仰 pitch(deg)	-0.00000001	0.00000007	0.00000015	-0.00000015
航向 yaw(deg)	0.00000014	0.00000015	0.00000021	0.00000006

由上述结果可以看出：经纬度误差精度为 $1.6 \times 10^{-7}$  deg 和  $4.5 \times 10^{-7}$  deg，满足  $10^{-7}$  deg 的精度要求；高程误差精度为  $0.03450490m$ ，满足 2m 的精度要求；速度误差精度分别为  $[0.713 \ 1.82 \ 6.105] \times 10^{-5} m/s$ ，满足  $10^{-5} \sim 10^{-4} m/s$  的精度要求；

姿态误差精度分别为 $[0.9 \ 0.7 \ 1.5] \times 10^{-7} \text{ deg}$ ，满足 $10^{-7} \text{ deg}$ 的精度要求。整体而言，自编程序解算结果与参考结果高度符合，说明解算程序基本正确。

进一步分析表明，系统误差曲线呈现典型的舒勒周期波动特征，验证了加速度计比力补偿与地球转动补偿逻辑的物理正确性；高程误差相较于水平分量表现出的量级差异，符合垂直通道不稳定性规律。

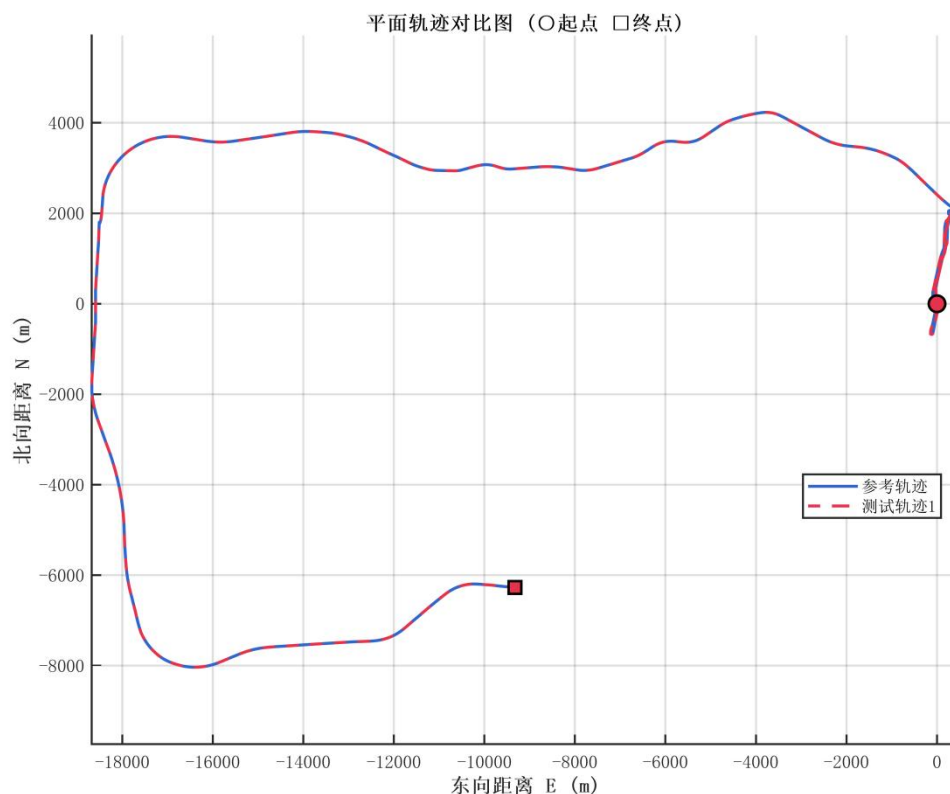


图 4.3 示例数据平面轨迹图

以初始位置为原点，绘制解算结果与参考结果的平面轨迹图（如图所示）。从图中可见，两者轨迹基本吻合，整体形态与走向具有良好一致性。然而，由于本实验所使用的参考轨迹绝对位置变化过大，推断其同样源于纯惯性导航解算结果，其本身并不具备绝对精度基准，因此该重合现象主要验证了程序设计与核心算法流程的正确性，而无法直接评估系统在绝对位置上的定位精度。

## 4.2 实测数据结果分析

与示例数据不同，实测数据的参考轨迹来源于高精度 GNSS/INS 紧组合解算结果。该系统融合了卫星观测值与惯导推算信息，具有严格的时空基准与误差控制模型，因此能够有效反映导航系统的绝对定位精度，为性能评估提供了可靠的参考基准。

下面直接给出未经零速修正的解算结果与参考结果之间的原始数据对比图

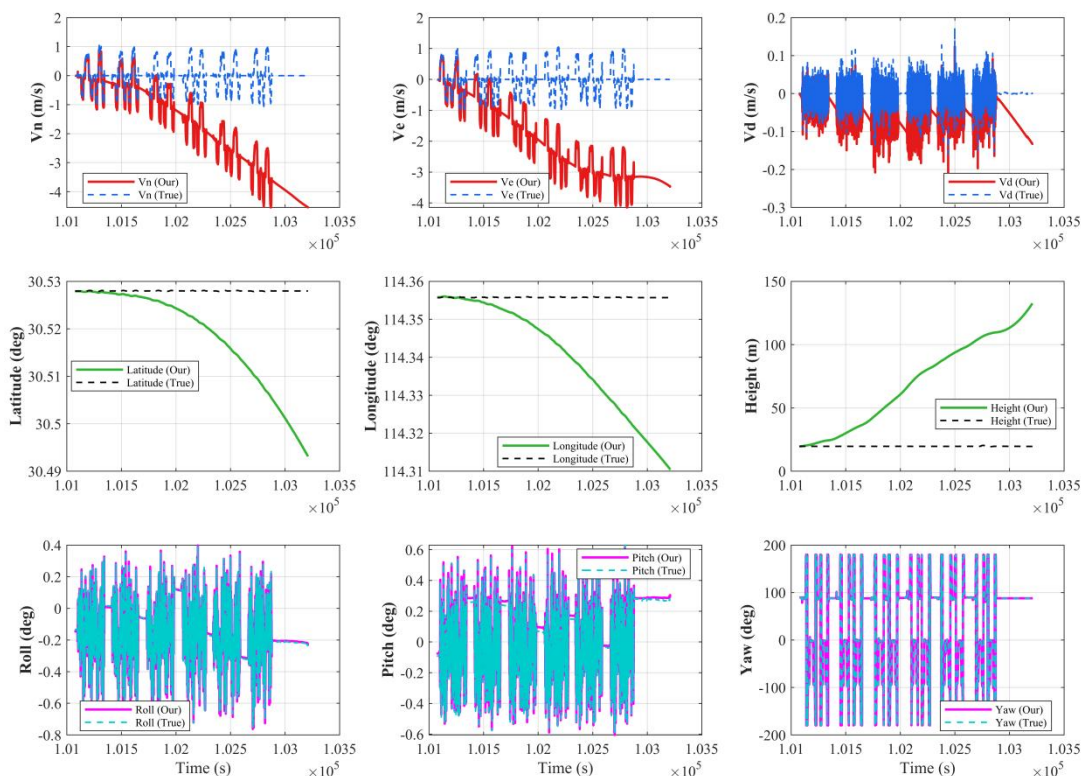


图 4.4 实测数据原始结果对比图

从图中可以看出：对于解算速度和位置在初始阶段基本与参考结果基本吻合，但随着时间推移逐渐向某一方向发散，这表明实验中并未对加速度计零偏进行有效补偿，导致零偏引起的速度误差随时间线性、位置误差随时间平方级增长；同时可以观察到，红色曲线表现出剧烈的震动形式，反映出 IMU 记录载体高频运动信息的能力，但由于缺乏速度修正，其均值严重偏离真实值，难以有效抑制误差积累；对于姿态角，相比速度和位置显然发散程度更小，这是因为姿态主要通过陀螺输出角速度一次积分得到，而陀螺误差量级通常远小于加速度计。

将解算结果与参考结果做对应时段的差分并统计均方差，可以得到下图

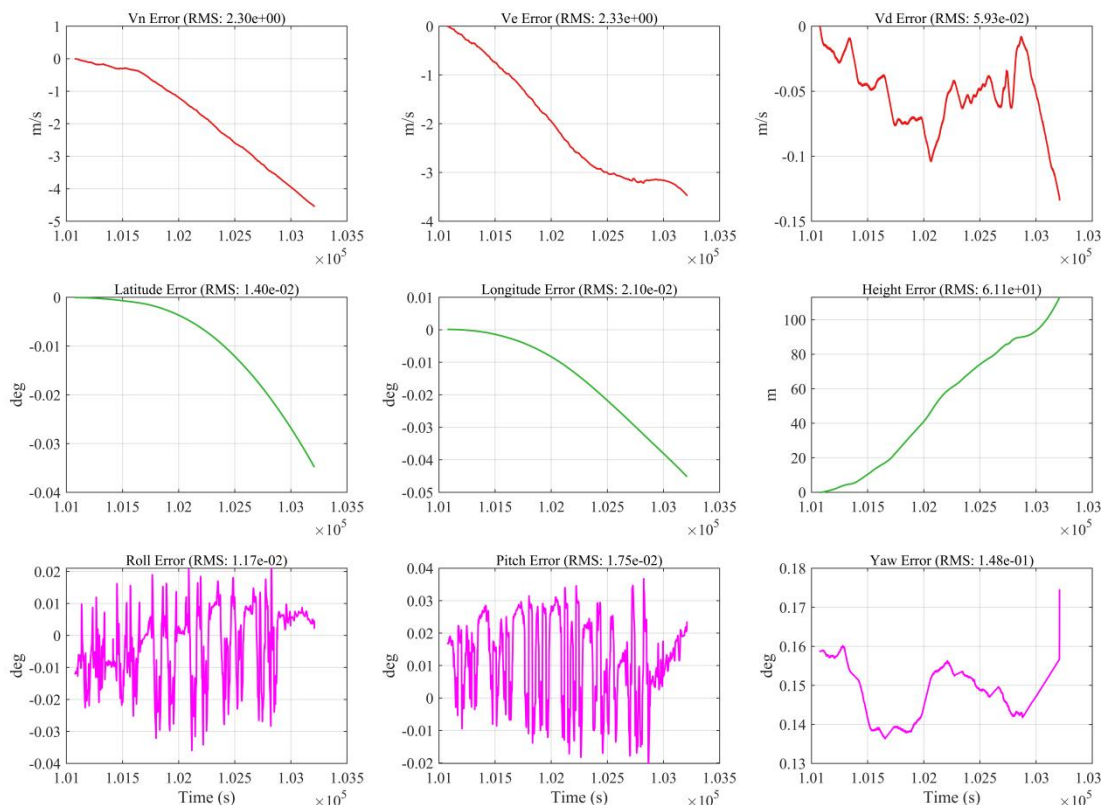


图 4.5 实测数据误差图

结合上述图像，，可以进一步分析得到：高程方向的速度误差(0.059m/s)明显小于东北方向(2.30m/s 和 2.33m/s)，分析原因，由于重力投影在水平方向造成的影响为正弦级而在垂向为余弦级，因此其对水平速度误差产生更大影响；对于位置误差，与前面分析结果基本一致，经纬度方向均方差为  $0.014^\circ$  和  $0.021^\circ$ ，而在高程方向发散明显，均方差达到了 61.1 米，说明纯惯导解算中高程呈现出不稳定性；横滚与俯仰的均方差均在  $0.01^\circ \sim 0.02^\circ$  量级。虽然曲线伴随高频震荡，但整体量级远小于航向角 ( $0.148^\circ$ )。这说明水平姿态在短时间内受重力约束较好，而航向角由于缺乏外部地磁或双天线辅助，完全依赖陀螺仪积分，受陀螺漂移的影响最为直接。

将上述结果转换到 NEU 方向并绘制平面轨迹图可以得到：

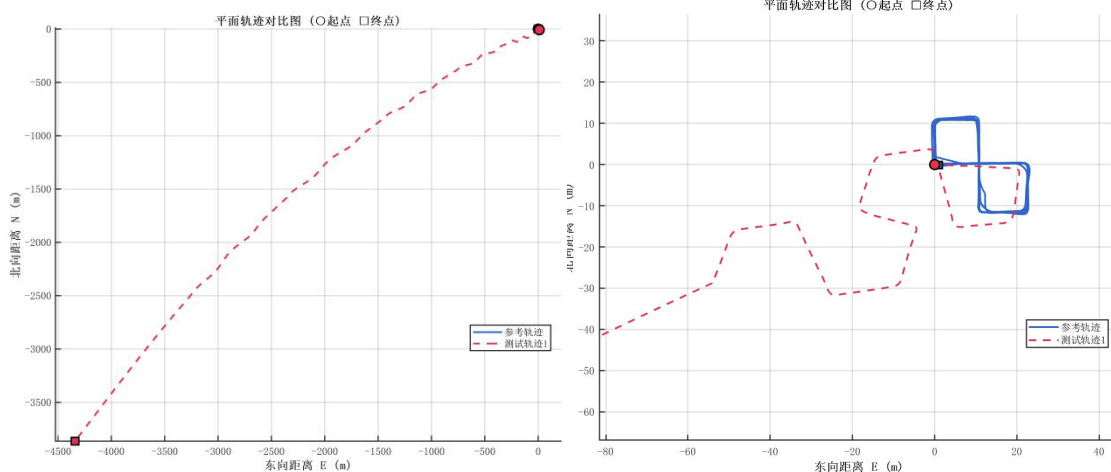


图 4.6 实测数据平面轨迹图

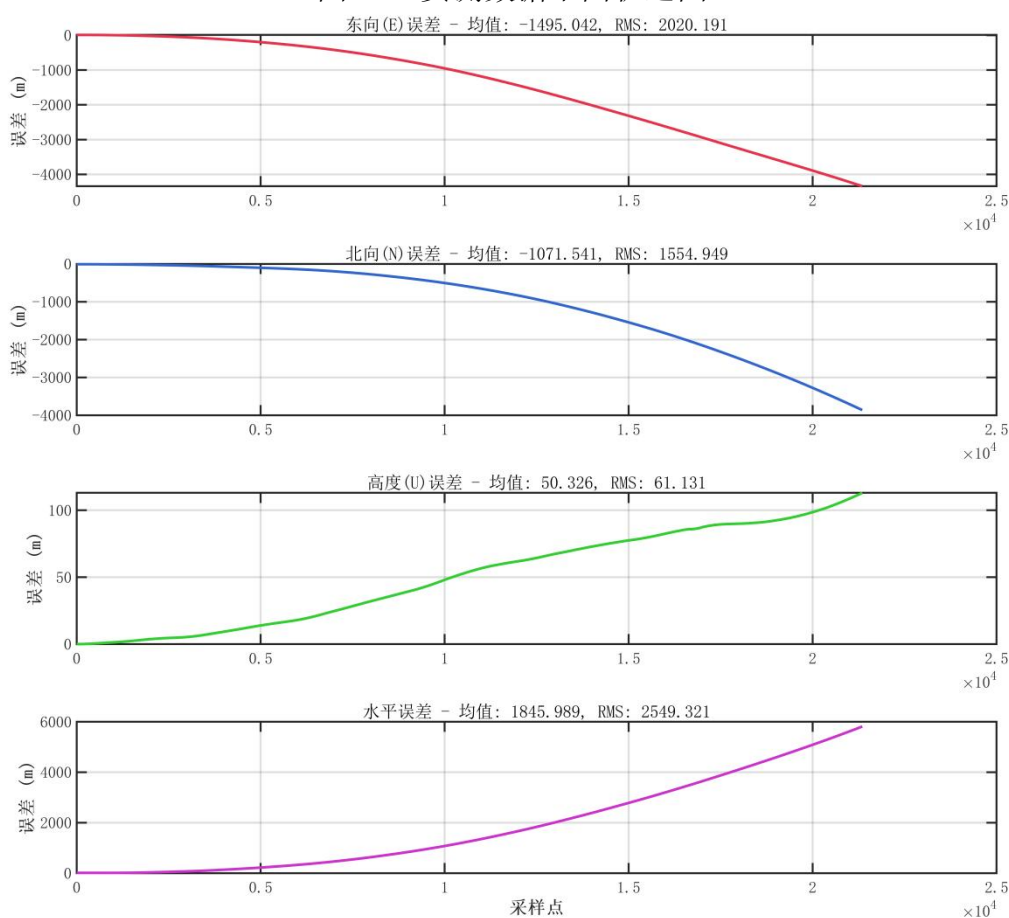


图 4.7 ENU 方向误差

转换为站心坐标系后的结果表明，系统误差随时间呈明显的二次方增长，水平总误差在实验结束时达到了 6 千米，说明加速度计零偏等误差在积分过程中的累积效应。由轨迹图可以看出，在起始历元中，惯导轨迹与参考轨迹较为符合，但随着时间推移，惯导解算结果迅速发散，使得解算结果产生严重的西南向伪运动，从而导致惯导的自主定位精度迅速失效。为降低误差，后续应对陀螺仪和加速度计标定，并采取零速修正等手段进行解算。



### 4.3 零速修正结果分析

本次实验中采取在如下时间段进行静止零速修正或者静态对准：

表 4.2 静止时间表

序号	时间段	总时间(s)	说明
1	100775~101098	323	静态粗对准
2	101347~101415	68	零速修正
3	101657~101732	75	零速修正
4	101986~102062	76	零速修正
5	102281~102341	60	零速修正
6	102592~102663	71	零速修正
7	102881~103210.54	329.54	粗对准

根据上述静止时间表，对实测数据进行零速修正，解算结果如下：

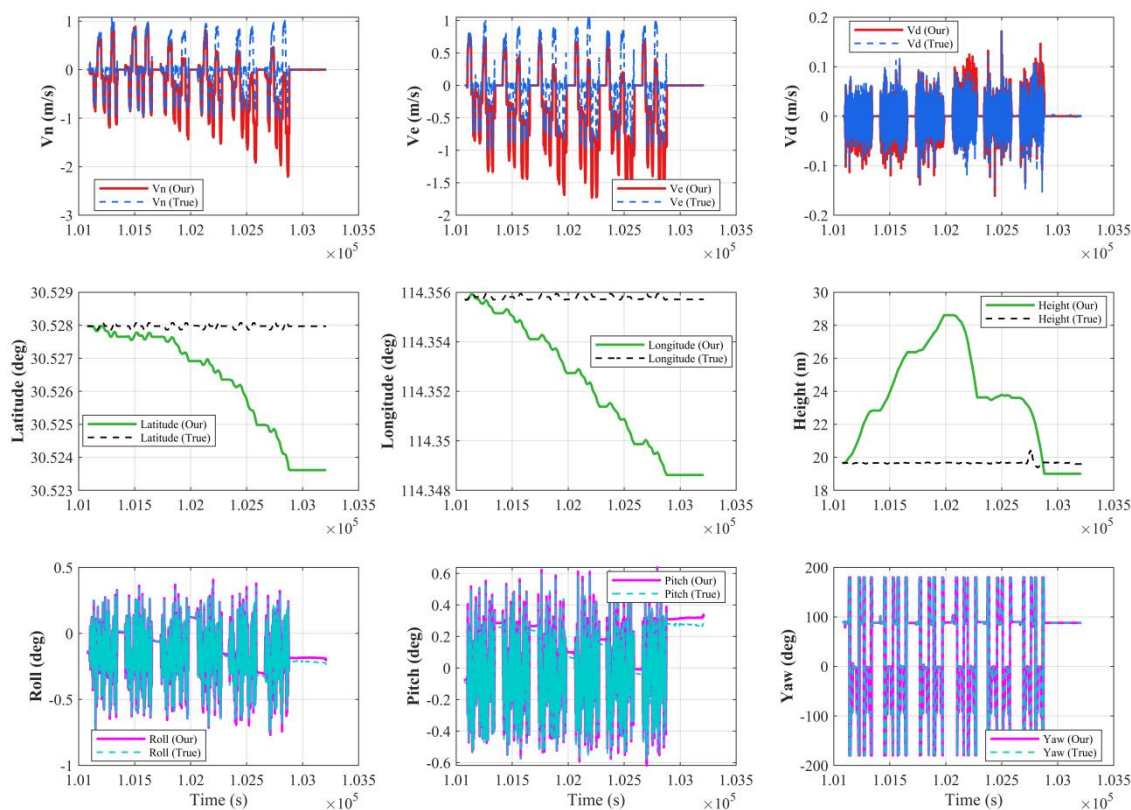


图 4.8 零速修正结果对比图

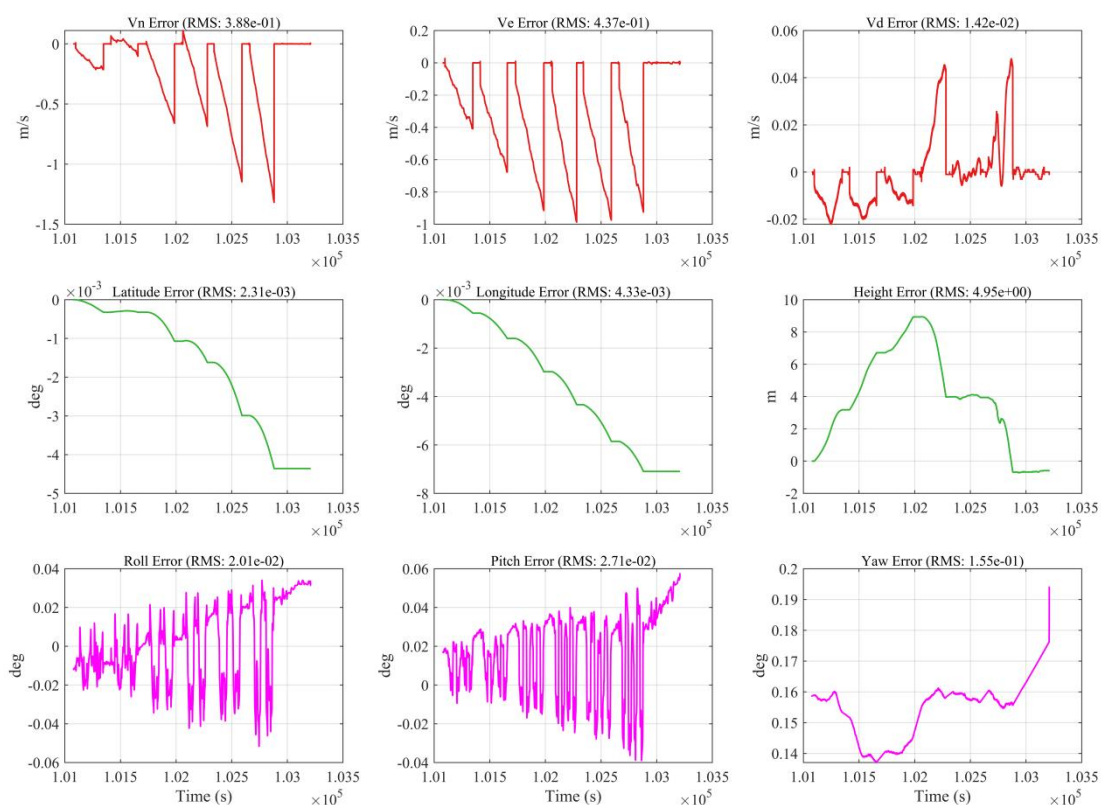


图 4.9 零速修正结果误差图

对比未经零速修正的结果，从上述结果可以看出：加入零速修正后，导航系统的性能大幅提升。速度误差不再随着时间线性发散，而是每隔一段时间强制清零重新累计发散，水平速度 RMS 从之前的约 2.3m/s 降至 0.4m/s 左右，精度提升了一个量级，垂向速度相对变化不大，但相比之前单向发散，现在分布更加均匀稳定；位置误差尽管仍随着时间非线性发散，但是引入零速修正后，速度在静止段归零，位置误差被周期性切断，在一定程度上减缓了位置误差的发散速度，经纬度误差从之前的  $0.04^\circ$  (约 4000 米) 骤降至约  $0.004^\circ$  (约 400 米)，高程误差从 61.1 米降至 4.95 米，精度显著提升；然而，零速修正对航向角的漂移抑制效果有限，这说明了航向误差在零速条件下仍然具备弱可观测性，因此，对于姿态角误差，必须通过外部其他条件进行修正。

将上述结果转换到站心坐标系下(如下图)，可以看出引入零速修正后，水平误差综合 RMS 从 6000 米降到了 488.3 米，高程误差 RMS 从 61.1 米降至 4.953 米，NEU 曲线表现出明显的阶梯状特征，即 ZUPT 在载体静止段有效抑制了速度误差的继续积累，并且可以看到高程曲线在最后阶段重新回到了零点附近，这说明 ZUPT 对抑制垂直通道不稳定性十分有效。与此同时，尽管平面轨迹仍然呈现出西南

方向漂移的趋势，但相比纯惯导轨迹，修正后的曲线已经能够辨认出实际运动轨迹，即整体精度上升了一个量级。放大平面轨迹初始段，可以发现，相比纯惯导初始轨迹，零速修正初始轨迹并无明显的改善，说明初始阶段主要受到仪器零偏等影响导出解算结果偏移，实际应用中需要加以改正。

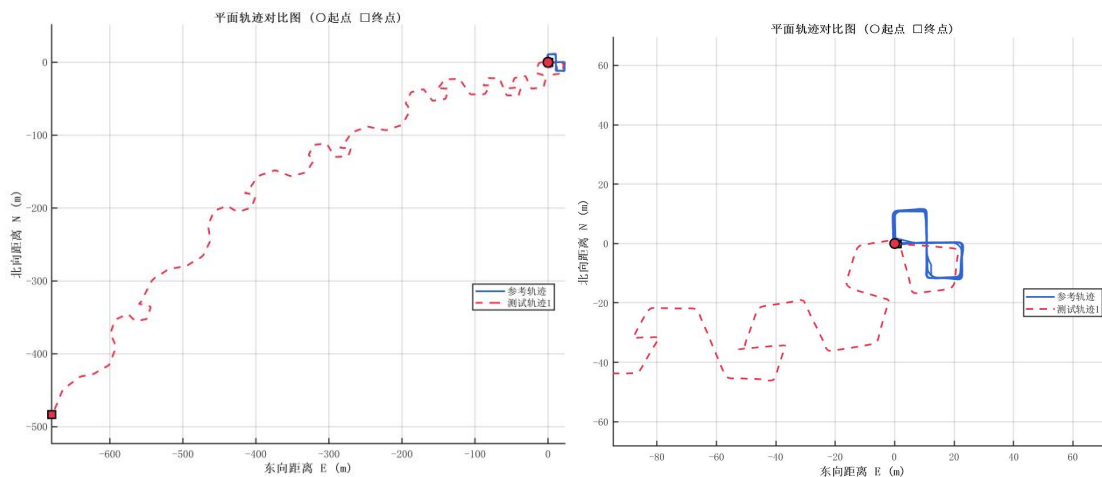


图 4.10 零速修正平面轨迹图

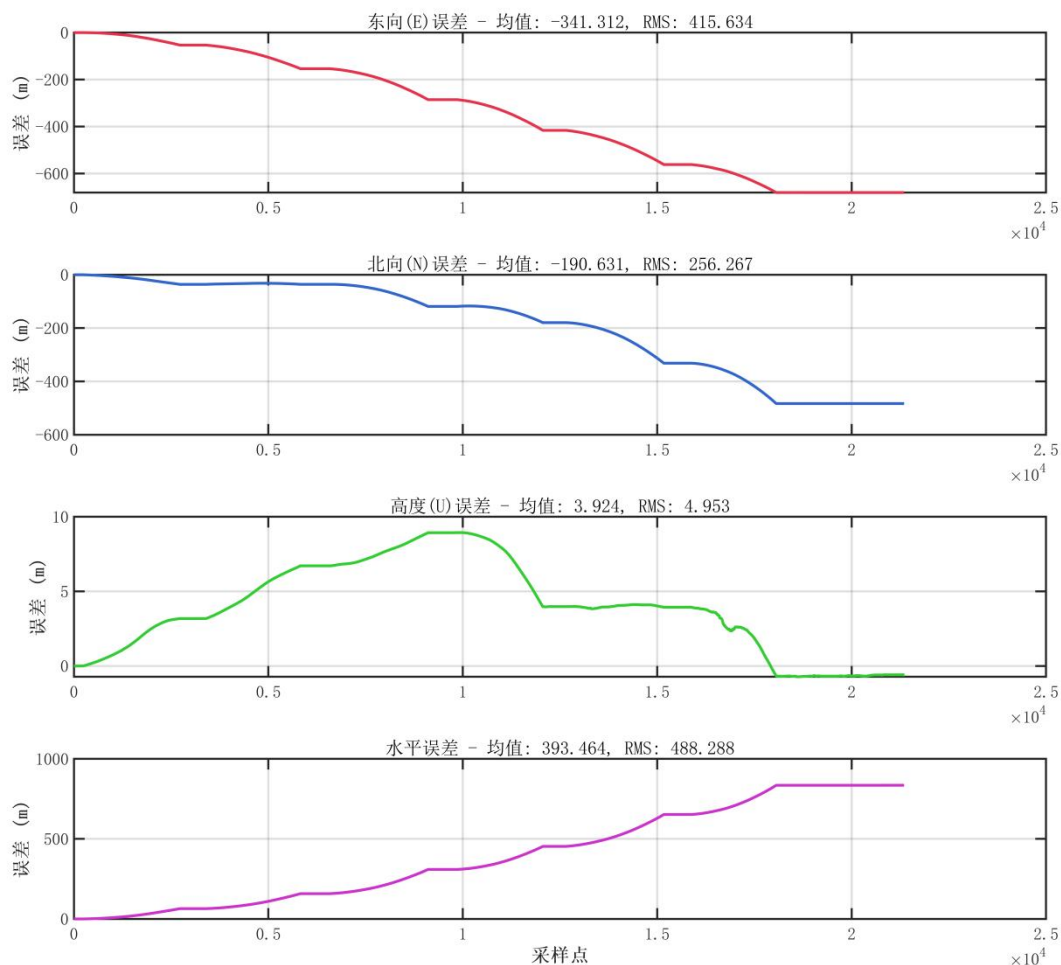


图 4.11 零速修正 ENU 方向误差



### 5.1 思考题

#### (1) 为什么用示例数据对比的差异很小而小推车的纯惯导误差却大很多？

根据参考结果的来源看，示例数据参考结果是课题通过纯惯导算法解算得到的动态导航结果，而非载体实际的运动结果，而在实验中，所提供的参考结果为 GNSS/INS 组合得到的精确结果，可以视为小推车的实际运动轨迹，所以使用示例数据对比差异比实际数据差异小。

#### (2) 尝试分析为什么纯惯导的高程误差累计速度比平面快？

尽管在实验条件下，平面误差绝对值大于高程绝对误差，但从系统稳定性和误差积累机制上来看，高程误差理论上更加不可控。

首先，根据高程的更新机制，一旦高程出现异常，系统会根据当前异常的高程值去计算得到异常的重力值，而后再根据这个异常的重力值得到异常的垂向速度，进而高程方向解算更加异常，因此，惯导系统在解算高程时，存在一个恶循环正反馈回路，一旦高程出现异常，就会迅速发散，难以从物理机制上将其拉回。

其次，由于高程方向比力测量存在重力的影响，如果重力模型存在一定的偏差，就会导致高程方向出现除零偏之外的系统误差，进而随着时间系统误差二次方发散产生更大的影响。

最后，惯导系统在水平方向存在着舒勒反馈机制。由于地球是圆的，当水平位置产生误差时，载体坐标系与当地地理坐标系会产生微小夹角，导致重力矢量泄露到水平方向，产生的力方向通常与误差方向相反，会引导系统产生周期的震荡，尽管不能消除误差，但它能抑制误差的无限制增长，使水平误差处于一定震荡范围内。而垂直通道则并没有该机制，而是垂向恶循环正反馈机制。

#### (3) 从数据融合的角度，如何更合理地使用零速修正信息？

从数据融合的角度出发，更合理的零速修正方案应该放弃简单的速度硬性赋零，而是构建 Kalman 滤波方程，通过将零速信息作为伪观测值输入滤波器，使得系统不仅能够修正瞬时速度误差，而且能够利用状态耦合在线估计出零偏等仪器误差。

其次，零速判断机制也需要综合多历元信息，而不是简单单历元幅值判断，建立更加健全的零速判断机制，防止误判漏判的可能性。

## 5.2 问题与思考

尽管本次惯导实验最终顺利完成，但在实验数据采集、程序编写以及数据分析的全过程中，并不是一帆风顺的，而是遇到了多层面的技术挑战与难题。

首先，针对数据处理方面，最为突出的问题是传感器的  $b$  系与算法定义  $b$  系之间的对齐偏差，针对这一问题，按照对应的轴系调整定义在数据解算前对数据预补偿即可。但是，由于本次实验与前两次实验采取的惯导型号有所区别，其输出数据格式存在一定差异，在最开始直接采用前面实验定义过的读取函数存在异常读取的问题，因此，本实验还在之前实验的基础上拓展了读取函数以解决该问题。

对于数据解算部分，由于解算途中存在频繁的姿态形式转换，引入了浮点数舍入误差，导致在实测某些历元中存在异常 `nan` 值解算情况，针对该问题，实验中尽可能采取单一姿态形式进行转换并时刻保持姿态四元数的正交性，在定义姿态函数时进行相应的容错处理除零保护，防止数学计算上的问题；对于零速判断部分，既要防止幅值阈值过大误判，也要防止漏判情况，因此，本实验采取双重保险，首先调试选取合适的幅值阈值，然后设置固定时间窗口综合判断。

## 5.3 总结

本实验通过小推车数据采集、捷联惯导解算、零速修正与解算结果分析等环节，完整地构建了一个自主导航系统的闭环处理流程。实验中通过示例数据解算结果与参考结果对比、无零偏修正解算结果与参考结果对比、零偏修正结果与参考结果对比以及零偏修正前后解算结果隐形对比，清晰地展示了纯惯导系统误差随着时间发散的固有特性，并验证了零偏、比例因子等误差的严重影响。通过对比引入 ZUPT 前后的误差曲线和平面轨迹图，还可以看到非完整性约束对抑制位置漂移的显著效能，即将定位精度提升了一个量级。

结合本学期的《惯性导航原理》课程的理论学习，本实验从观测测量器件的原理到完整惯性导航系统的集成实现，全面加深了我对惯性导航工作机制的理解。特别是在传感器标定、静态粗对准、机械编排等关键环节，通过实验程序编写，使我更直观地认识到了理论公式背后的物理意义以及在工程实践中可能面临的挑战。此外，课程学习与实验也开拓了我的眼界，对我的科研思维产生了深刻影响，通过理论与实践的逐步学习，逐渐养成了从零开始开发算法程序的能力思想。

针对本课程的教学与实验环节，整体难度适中，教导循序渐进，符合初学者的

思维，但为了课程发展可以精益求精，个人提出以下建议：首先，本课程中进行的惯导误差标定、静态粗对准与动态纯惯导定位三个实验，涵盖惯性测量单元从出厂到应用的全过程，但是由于三次实验采取的是不同型号的惯导，因此实验相对独立，个人认为可以将三者进行融合，一来可以改正纯惯导中出现的漂移现象，其次还可以从一个更全面的视角回顾整个课程的学习；其次，针对实测数据解算结果，第一次看到会感觉十分诧异，因为其极其不符合预期轨迹，会不由自主地怀疑自己程序解算的问题，因此，课程实习中可以针对该实验给出一个大致的结果以方便没有经验的初学者判断实验错误大致出在什么方面。

最后，衷心感谢朱峰老师在课程讲授中的深入讲解与耐心指导，感谢张万威老师与孙骁老师在实验环节的全程陪伴与辛勤付出，同时也感谢助教师兄在数据提供与结果比对中给予的支持与帮助。

## 参考文献

- [1] 武汉大学测绘学院. 惯性导航课程讲义[Z]. 武汉：武汉大学，2023.
- [2] 牛小骥，陈起金，张全. 惯性导航原理与算法设计[M]. 武汉：武汉大学卫星导航定位技术研究中心，出版年.
- [3] KF-GINS. INS/GNSS 组合导航开源软件[EB/OL]. <https://github.com/i2Nav-WHU/KF-GINS>, 2023-12-25.