Researching and building a method for testing and adjusting the strapdown inertial navigation system

Nguyen Trong Khuyen*, Vu Hai Ha, Ho Si Duong, Cao Duc Sang

The Control Automation in Production and Improvement of Technology Institute.
*Corresponding author: nguyentk126@gmail.com.
Received 8 September 2021; Revised 1 December 2021; Accepted 15 December 2021.
DOI: https://doi.org/10.54939/1859-1043.j.mst.csce5.2021.24-31

ABSTRACT

The strap-down inertial navigation system (SINS) is widely used and becoming very important in many areas, especially in the arms industry when the GPS signal is lost or not reliable. To ensure the precision of the system, in addition to optimizing the algorithm for the strap-down inertial navigation system, the testing, and adjusting of the SINS system when installed on vehicles also play a vital role. In the article, a method of displaying SINS data on a digital map is proposed. Furthermore, the article also proposes a method to assess the influence of misalignment angles on the navigation accuracy and how to estimate and correct them.

Keywords: Strapdown inertial navigation system (SINS); Digital map; Misalignment angles; PSO algorithm.

1. INTRODUCTION

To accelerate the process of testing and adjusting the inertial navigation system, it is necessary to display navigation data [3] on a digital map. Domestic uses of digital maps mostly focus on making use of the integrated devices available on the market, which have a built-in map and use satellite navigation system (GPS). Local research on building up a digital map used for the inertial navigation is still very limited. To address the requirement for testing the SINS system, the authors propose a method of displaying the inertial navigation data on digital maps based on Gmap [6], and build a map software to serve the testing process.

Another factor, which directly influences the quality of SINS is the misalignment angles [3, 4] (misaligned roll - $\Delta\psi_0$, misaligned pitch - $\Delta\theta_0$, misaligned yaw - $\Delta\phi_0$) between the inertial navigation system module and the carrying vehicle. When installing the inertial navigation system module on the carrying vehicle, we expect these misalignment angles zeroed, but in fact, the misalignment is inevitable. To tacle this issue, together with proposing a method of integarating a digital map, the article proposes research of the effect of misalignment angles on navigation error, how to estimate and correct them.

2. A METHOD OF DISPLAYING NAVIGATION DATA ON DIGITAL MAP

2.1. The overview

In the test process, the digital map is used to display navigation trajectories. In addition, it is also used to show other sources of data for references and adjustment, such as GPS data, benchmark positions,... Figure 1 shows how to put data on a digital map.

Figure 1. *The block diagram of displaying data on a map.*

Sources of input data: Include data returned from inertial navigation module, global positioning system (GPS) module, and data entered by users (including positions of reference, marked positions of stores, bridges, port,…).

Module reads and decodes data: The data from an inertial navigation system or GPS module are packed and protected by checksum code and must be synchronized and decrypted. The results of decoding consist of different fields (longitude, latitude, altitude, etc.).

Data management and display output module: The decoded data are structured into different fields (benchmark field, INS orbital field, GPS orbital field,…) and can be stored as a file and retrieved through address. The module also creates data streams to connect to the map interface.

Map interface: Displays navigation trajectories, gives users an intuitive view.

2.2. Solutions updating data on digital map

The solution using map-integrated devices: There are many devices on the market, which have a built-in digital map, using satellite data as a source of navigation. To update data onto the map, we need to convert packets of inertial navigation data into satellite navigation data ones before feeding them to the devices. This method also asks to change to the hard device. Furthermore, the parameters on display are limited by the features of the devices.

The solution building digital map on the .net framework platform, using C# language, Gmap library: This method solves the disadvantages of the first one and is used by the authors in practice for the test. Designing and building software on the Gmap platform gives high customization, allow the authors to customize the map. Figure 2 shows the inertial and satellite navigation trajectories on the digital map. Comparing the INS trajectory (red line) with the GPS trajectory (blue line) allows us to make a subjective assessment of the quality of the navigation system.

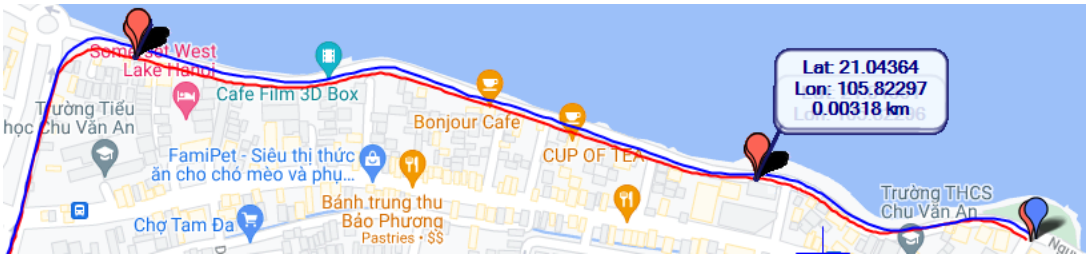

Figure 2. *Display of navigation trajectories on digital maps (map scale 1:4000).*

In order to evaluate the error of the inertial navigation system quantitatively, it is necessary to do data statistics when running the test. The evaluation is done by comparing the coordinates returned by the inertial navigation module with the reference coordinates. The difference between these two coordinates is recorded after a serial of tests, and then statistics are performed. The navigation error is the mean squared deviation of the mentioned coordinates.

3. ASSESSMENT AND ADJUSTMENT OF EFFECT OF MISALIGNMENT ANGLES ON NAVIGATION ERROR

3.1. Clarification of navigation error caused by misalignment angles

In the NED coordinate system, the kinematic model of the inertial navigation system is:

$$\dot{v}^n = C_b^n f^b - (2\omega_{ie}^n + \omega_{en}^n) \times v^n + g^n$$

$$\dot{r} = \begin{bmatrix} \frac{1}{R_M + h} & 0 & 0 \\ 0 & \frac{1}{R_N + h} & 0 \\ 0 & 0 & -1 \end{bmatrix} v^n = R_c . v^n \quad (1)$$

Where: ω_{ie}^n is the rotational velocity of the earth's rotation relative to the inertial reference system with coordinates presented in the coordinate navigation system; ω_{en}^n is the rotational velocity of the coordinate navigation system relative to the earth coordinate system with coordinates presented in the coordinate navigation system; f^b is the acceleration measured by the accelerometer in the body coordinate system associated with the IMU; g^n is the acceleration of gravity presented in the coordinate navigation system; C_b^n is the transformation matrix from the body coordinate system to the coordinate navigation system; h is altitude; RM, RN are respectively radiuses of curvature of latitude and longitude.

By using discrete recursion, it is possible to determine the navigation coordinates according to the formula (1). However, in order to increase the accuracy of the system, the kalman filter is applied in the navigation algorithm to carry out the correction. The measured value entered into the Kalman filter is the speed $v^{n'}$ inferred from the velocity sensor mounted on the vehicle:

$$v^{n'} = C_{b'}^b v^{b'}$$
$$v^{b'} = \begin{bmatrix} V_{odo} & 0 & 0 \end{bmatrix} \quad (2)$$

V_{odo} - Velocity measured by sensor mounted on the carrying vehicle.

$C_b^{b'}$ - Transformation matrix [5] from the body coordinate system to the system associated with vehicle. This matrix is inferred from misalignment angles:

$$C_b^{b'} = \begin{bmatrix} \cos(\Delta\psi_0) & -\sin(\Delta\psi_0) & 0 \\ \sin(\Delta\psi_0) & \cos(\Delta\psi_0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\Delta\theta_0) & 0 & \sin(\Delta\theta_0) \\ 0 & 1 & 0 \\ -\sin(\Delta\theta_0) & 0 & \cos(\Delta\theta_0) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\Delta\Phi_0) & -\sin(\Delta\Phi_0) \\ 0 & \sin(\Delta\Phi_0) & \cos(\Delta\Phi_0) \end{bmatrix} \quad (3)$$

In Formula (2), because we do not know exactly the misalignment angles and these angles are usually very small, so in calculation, the matrix $C_b^{b'}$ is set to the unit matrix (I), and this is the main cause of the navigation error caused by the misalignment angles.

If the misalignment angles are well known, we could carry out the mechanical correction of axes of the inertial navigation module to adjust the misalignment angles to zero. Another way is to use formulas (2) and (3) to carry out a correction in the software.

3.2. Assessment of misalignment angles based on PSO algorithm

The PSO algorithm [7] was proposed by Kennedy and Eberhard and is used to solve many global optimization problems. The algorithm is based on the simulation of the collective intelligence of particles in the swarm to find the optimal solution in a search space. There are exchanges of information between the particles in a swarm to update the optimal position. In this algorithm, a way of movement in the search space is given to all particles so that after a finite number of steps, it is possible to find the optimal global position.

In the global optimization problem, the task is to find the position where the objective function minimizes. This position is called the optimal global position of the function.

$$F(X) = F(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, ...),\ X = (\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, ...) \in D \tag{4}$$

Where: $F(X)$ is the objective function; $X = (\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, ...)$ - arguments of the objective functions, restricted in a search space *D*.

3.2.1. Application of PSO algorithm

To apply the PSO algorithm, we have the following definitions:

- The search space *D*: Is the set of angular vectors $X = (\Delta\psi, \Delta\theta, \Delta\varphi)$, with the condition:

$$-\varepsilon_1 < \Delta\psi < \varepsilon_1;\ -\varepsilon_2 < \Delta\theta < \varepsilon_2;\ -\varepsilon_3 < \Delta\varphi < \varepsilon_3 \tag{5}$$

Because the misalignment angles are usually very small, in the calculation, we could choose $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = 0.1$ radian (5^0), in practice, these values could be smaller, about 1^0 (0.02 radian).

- The objective function F(X): With each X vector and the same set of IMU simulation data (collected when the vehicle runs from a reference point A to point B (x_b, y_b, h_b)) we run the navigation algorithm at the end we obtain a coordinate B' (x'$_b$, y'$_b$, h'$_b$). The value of objective function F(X) is the distance between B and B':

$$F(X) = BB' = \sqrt{(x_b - x_b')^2 + (y_b - y_b')^2 + (h_b - h_b')^2} \tag{6}$$

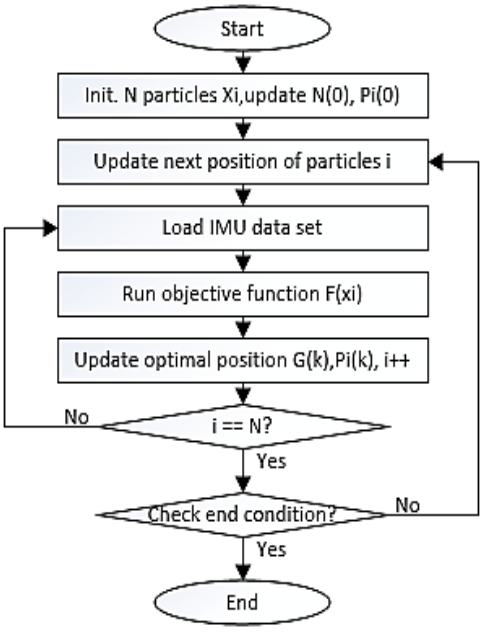

Figure 3. *PSO algorithm implementation flowchart.*

The aim of correction is to find a vector $X_0 = (\Delta\psi_0, \Delta\theta_0, \Delta\varphi_0)$ in order to $F(X_0)$ reaching the minimum. From this point of view, the PSO algorithm can be used to solve this vector. The X_0 vector, naturally, considers as the true misalignment angle vector. The PSO algorithm implementation is shown in figure 3.

Step 1: Initialize N particles (N = 16, 32, 50…) with position vectors $X_i(0) = (\Delta\psi_{i0}, \Delta\theta_{i0}, \Delta\varphi_{i0})$ in search space D. These vectors can be initialized randomly around the predefined value. The global optimal position G(0) at the starting point is one of vectors $X_i(0)$, where $F(X_i(0))$ has the smallest value; the optimal local position of the i-th particle $P_i(0) = X_i(0)$. Vector $v_i(0)$ is called the velocity of the i-th particle at the beginning, by default, $v_i(0) = 0$.

Step 2: Update the positions of the particles: Suppose that in the k-th loop, the i-th particle has position Xi(k). Put G(k) - The optimal global position and Pi(k) - The optimal local position of the i-th particle. In the (k + 1)-th loop, the positions of i-th particle is determined by the formular:

$$\begin{aligned} v_i(k+1) &= w.v_i(k) + r_1c_1(P_i(k) - X_i(k)) + r_2c_2(G(k) - X_i(k)) \\ X_i(k+1) &= X_i(k) + v_i(k+1) \end{aligned} \tag{7}$$

Where w – Inertia weigh, c_1, c_2 acceleration coefficients, r_1, r_2 – Random values in the range (0, 1).

Step 3: Update the optimal positions: There are exchanges of information between the particles to update the optimal global position G(k) and the optimal local position Pi(k) after each iteration. The updating process is performed for all particles and follows the algorithm:

$$\begin{aligned} &if\,(F(X_i(k+1)) < F(G(k))):\ G(k+1) = X_i(k+1) \\ &else: G(k+1) = G(k) \\ &if\,(F(X_i(k+1)) < F(P_i(k))):\ P_i(k+1) = X_i(k+1) \\ &else: P_i(k+1) = P_i(k) \end{aligned} \tag{8}$$

Step 4: End the search process: The positions G(k) will gradually converge to the optimal global position, the particles also converge to this position. We suppose that after L iterations when G(k) converges to a neighborhood of the point $X_0 = (\Delta\psi_0, \Delta\theta_0, \Delta\varphi_0)$ with a small enough error ε, the program can be terminated. Vector X_0 can be seen as an approximate representation of the actual misalignment angle vector with the error ε.

3.2.2. Simulation on Matlab

Let the vehicle run from position A with coordinate (21.031630, 105.840420, 4 m) to position B (21.040470, 105.842030, 4.1 m) and put B' - coordinate returned by the INS module when the vehicle stops at position B. Make a collection of data returned by IMU, which will be used to run on Matlab. Run PSO algorithm according to the flowchart in Figure 3. For each misalignment angle vector $X = (\Delta\psi, \Delta\theta, \Delta\varphi)$, we use formulas (2), (3) to calculate the transformation matrix $C_b^{b'}$ and run the inertial navigation algorithm with the IMU collected data. The returned value of the objective function is the distance

between the coordinate B' returned by the navigation module and the reference coordinate B, F(X) = ||B-B'||. The results of running PSO algorithm on Matlab give us the misalignment angle vector X0 = (-0.002, 0.002, 0.007) radian (see Figure 5).

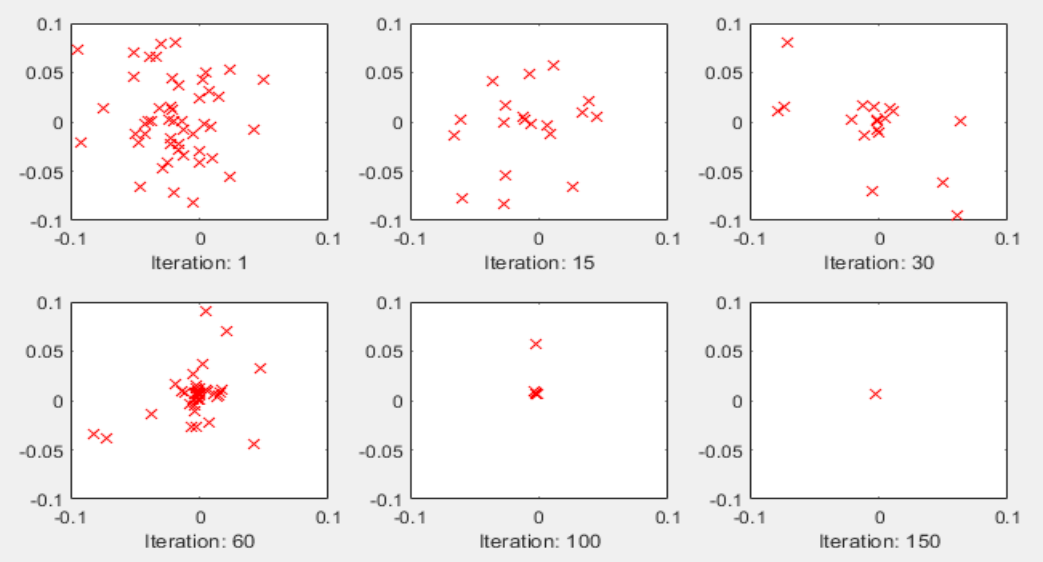

Figure 4. *The convergence of particles to the global optimal position.*

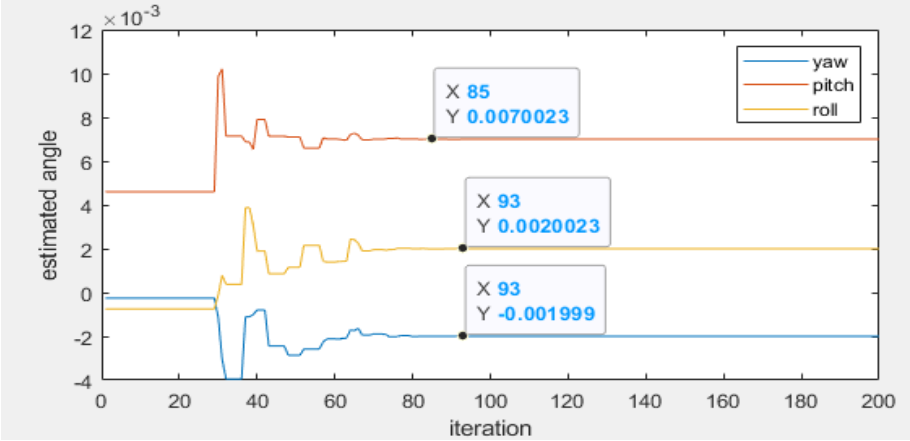

Figure 5. *The convergence of G(k) to the global optimal position (true misalignment angles).*

In the simulation, the search space D is restricted by the condition $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = 0.1$. The inertia weight w = 0.98, acceleration coefficients c_1 = c_2 = 2, the number of initial particles is 32 and the number of iterations is 200. From the result in Figure 5 we can see that the particles start converging to the optimal global position at iteration of 80, the optimal positions after each iteration also converge to this optimal position.

3.3. Results of correction

Let the vehicle run along Ly Nam De street. The test results before and after correcting the misalignment angles X0 = (-0.002, 0.002, 0.007) (radian) are shown in Figure 6 and Table 1. It is obvious that the navigation accuracy is significantly improved after correction.

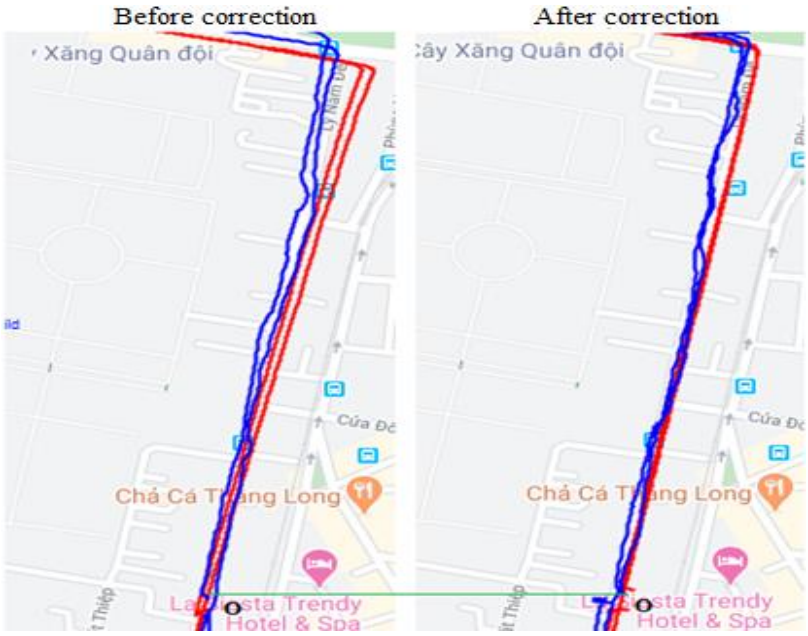

Figure 6. *Vehicle trajectories before and after misalignment angles correction.*

Table 1. *The difference between INS and GPS coordinates before and after correction of misalignment angles.*

Coordinates before correction (Lat, Lon)			**Coordinates after correction (Lat, Lon)**		
GPS	**INS**	**Deviation (m)**	**GPS**	**INS**	**Deviation (m)**
(21.03208^0, 105.84435^0)	(21.03207^0, 105.84437^0)	2.10	(21.03208^0, 105.84435^0)	(21.03207^0, 105.844375^0)	2.11
(21.03484^0, 105.84495^0)	(21.03483^0, 105.84498^0)	4.24	(21.03484^0, 105.84496^0)	(21.034835^0, 105.84498^0)	4.10
(21.03715^0, 105.8454^0)	(21.03714^0, 105.84455^0)	10.61	(21.03715^0, 105.84539^0)	(21.03715^0, 105.84452^0)	6.2
(21.03983^0, 105.84593^0)	(21.03977^0, 105.84606^0)	15.63	(21.03984^0, 105.84593^0)	(21.03982^0, 105.84602^0)	6.4

4. CONCLUSION

Thus, the article has come up with an effective solution to test and adjust the strap-down inertial navigation system. The article proposes a method to update navigation data on a digital map to make a visual evaluation of the system and presents a method to evaluate and correct the influence of the misalignment angles on navigation accuracy. The application of the PSO optimization algorithm is an extensible solution to estimate the optimal value of several other parameters affecting the quality of the inertial navigation system.

REFERENCES

[1]. Paul G. Savage, "*Down-summing rotation vectors for strapdown attitude updating*", WBN-14019, July 16, 2017.

[2]. Paul G. Savage, "*Strapdown Inertial Navigation Integration Algorithm Design Part 2: Velocity and Position Algorithms*", Strapdown Associates, Inc., Maple Plain, Minnesota 55359, Journal of guidance, control and dynamucs, Vol. 27, No. 2, March–April 2004.

[3]. Quân chủng hải quân, cục kỹ thuật, "*Hệ thống dẫn đường – Hành trình mạng tô pô và định hướng АЗИМУТ-БИ – Hướng dẫn sử dụng*", ПИКВ.462118.004 РЭ (in Vietnamese).
[4]. Haifeng Xing *ID*, Zhiyong Chen, Haotian Yang, Chengbin Wang, Zhihui Lin and Meifeng Guo, "*Self-Alignment MEMS IMU Method Based on the Rotation Modulation Technique on a Swing Base*", *Sensors*, 2018.
[5]. *https://en.wikipedia.org/wiki/Rotation_matrix*
[6]. Google Maps, "*Google Maps Tutorialpoints*".
[7]. Jaco F. Schutte, "*The Particle Swarm Optimization Algorithm*", EGM 6365 - Structural Optimization Fall 2005.

TÓM TẮT

NGHIÊN CỨU XÂY DỰNG GIẢI PHÁP KIỂM TRA HIỆU CHỈNH HỆ THỐNG DẪN ĐƯỜNG QUÁN TÍNH KHÔNG ĐẾ

Ngày nay hệ thống dẫn đường quán tính không đế đang được sử dụng rộng rãi và trở nên không thể thiếu trong nhiều lĩnh vực, đặc biệt là trong quân sự, trong điều kiện tác chiến bị mất tín hiệu GPS. Để đảm bảo độ chính xác của hệ thống dẫn đường quán tính thì ngoài việc tối ưu thuật toán cho khối dẫn đường, việc kiểm tra hiệu chỉnh hệ thống hệ thống khi lắp đặt và chạy thử nghiệm cũng đóng một vai trò hết sức quan trọng. Trong bài báo này, tác giả đưa ra giải pháp đưa dữ liệu dẫn đường quán tính lên bản đồ số khi chạy thử nghiệm và nghiên cứu đánh giá ảnh hưởng của góc lệch trục (giữa khối dẫn đường và phương tiện mang) lên chất lượng dẫn đường, phương pháp hiệu chỉnh để giảm sai số.

Từ khóa: Dẫn đường quán tính không đế; Bản đồ số; Góc lệch trục; Thuật toán PSO.