INS DATA PROCESSING AND APPLICATION IN DRONE BALANCE CONTROL

NGO THANH BINH, NGUYEN CANH MINH, LE MINH TUAN

PhD, Faculty of Electrical and Electronic Engineering,
University of Transport and Communications, Hanoi, Vietnam
Corresponding author's email: ngobinh74@utc.edu.vn

Summary: *This article introduces an INS data processing solution with an improvement of DCM algorithm using a MEMS INS 9-DOF sensor for vibration measurement and control. The significant feature of this system is the new method created to process data for device mounted on moving objects. This method is programed to self-correct the drifting data of INS at the measurement level based on compensating for tilt effect and development of a DCM using flexible PI. Anti-drift data solution enables 9-DOF INS system with capability of data self-correction to operate independently. From these research results, we applied successfully to get more efficient in balance of drones.*

Keywords: *INS data processing, DCM algorithm, Vibration measurement, Drone balance control*

I. INTRODUCTION

The two basic errors of INS (Inertial Navigation System) are drift and bias. Handling these two parameters is one of the key factors to solve accumulation error to improve the quality of INS. In addition, tilt angles, especially yaw angle affected by hard iron and soft iron, should be calibrated by accelerometers and magnetometers to compensate before using [2]. In the previous time, solutions in designing devices using INS was processed data in a way that had to accept drift and bias accumulative errors, and then use the Kalman filter or other compensation methods to adjust results [4], [5], [6]. This is negative solution, in which accuracy of result depended on many elements such as model of state and measurement, type of Kalman filer used. Using DCM (Direction Cosine Matrix) in recalculating elements of rotation matrix (R) is one of positive solutions, in which tilt angles are corrected at measurement level. In some DCM theories before, such as Phuong [1], Sergiu and William [3], they use INS 6-DOF combined with an extra compass or GPS to recalculate INS errors. In case of losing GPS signal, the device will lose heading data. Thus, the updated parameters of elements of R from updated omega values of their DCM algorithm are broken. Moreover, speed of receiving heading data of GPS is slow and data are not very accurate because its antenna is not always parallel with G reference during the movement time of moving object.

In this article, after testing to have parameters of PI (Proportional Integral), and to solve problems of the impact of elevation, bank angle, hard and soft iron effects on heading calculations, we introduce a formula to calculate yaw angle from 3-axes magnetometer integrated inside INS, and improve a solution to calculate DCM based on flexible PI, in which parameters of PI can be changed depending on errors of real tilt angles. Using this DCM algorithm, a MEMS INS 9-DOF can operate independently. From these results, we applied to control the movement of a drone for better horizontal balance, resulting in better shots and videos. This makes image processing easier in the next research step.

II. IMPROVEMENT OF DCM ALGORITHM USING MEMS INS 9-DOF

The rotation matrix R based on roll (ϕ), pitch (θ) and yaw (ψ) angles describes the orientation of a coordinate system with respect to another. A vector in one system can be transformed by multiply with this rotation matrix. To describe the motion of the object, we need to determine the status of the object compared to a suitable reference system, usually using the earth coordinate system (e-frame) or G (Globe/Ground). The orientation of an object is often described by three consecutive rotations, ordered according to the coordinate axes. Results of moving objects show in the body frame reference (b-frame) or P (Plane) compared to the reference frame G. This matrix is also called DCM because it consists of cosines of angles of all possible combinations of body and global vectors [1]:

$$R = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \quad (1)$$

The orientation of a moving object P reference with respect to G reference can be determined by using the two right-handed coordinate systems.

$$r(t) = r(0) + \int_0^t r(\tau) \times d\theta(\tau), \text{ with: } d\theta(\tau) = \omega(\tau)d\tau \quad (2)$$

Where: $r(0)$: The initial value of the vector

$\int_0^t d\theta(\tau) \times r(\tau)$: changing values in the vector

Processing them according to the row/column of R and considering them as rotation vectors, the best estimate of the true rotation rate is calculated as follows:

$$\omega(t) = \omega_{Gyro}(t) + \omega_{Cor}(t) \quad (3)$$

Where: $\omega_{Gyro}(t)$ is calculated angle velocities of gyros following three axes

$\omega_{Cor}(t)$ is gyro correction, ω_Correction

The rotation error between compass and the projection on the horizontal P of roll axis (X) of INS is an indication of the amount of drift. The correction rotation is the Z component of the cross product of the X column of the R matrix and the COG (Course Over Ground) vector. Using a 9 - DOF INS, COG of P reference with respect to G reference according to the axes are calculated by radiated to the coordinate axis to create formula:

$$Mag_x^G = Mag_x^P \cdot \cos\theta + Mag_y^P \cdot \sin\phi + Mag_z^P \cdot \cos\phi\sin\theta$$

$$\mathrm{Mag}_y^G = \mathrm{Mag}_y^P \cdot \cos\phi - \mathrm{Mag}_z^P \cdot \sin\phi \qquad (4)$$

$$\mathrm{Mag}_{\mathrm{Heading}}^G = \mathrm{atan2}(-\mathrm{Mag}_y^P, \mathrm{Mag}_x^P)$$

Where: $\mathrm{Mag}_{\mathrm{Heading}}^G$ is COG value in the earth frame of reference (G).

The reference vector from the normalized horizontal velocity vector can be calculated by taking the cosine and sine of the COG in the G reference, as below:

$$\mathrm{COGX} = \mathrm{Mag}_{\mathrm{Heading\_x}}^G = \cos(\mathrm{Mag}_{\mathrm{Heading}}^G) \qquad (5)$$

$$\mathrm{COGY} = \mathrm{Mag}_{\mathrm{Heading\_y}}^G = \sin\left(\mathrm{Mag}_{\mathrm{Heading}}^G\right)$$

So, we have yaw angle correction as following equations:

$$\mathrm{Yaw}(\mathrm{Mag}_{\mathrm{Heading}}^G) = r_{xx}\mathrm{COGY} - r_{yx}\mathrm{COGX} \qquad (6)$$

$$\mathrm{Yaw}\left(\mathrm{Mag}_{\mathrm{Heading}}^P\right) = \mathrm{Yaw}\left(\mathrm{Mag}_{\mathrm{Heading}}^G\right)[r_{zx} \quad r_{zy} \quad r_{zz}]^T$$

Accelerometers are used to correct roll-pitch drift errors. The centrifugal acceleration A_{Cen} in the P (b-frame) is calculated by the cross product of the gyro vector and the velocity vector. In the NWU coordinate system, the output data of the accelerometers are the result of gravity minus the acceleration. The reference measurement of gravity in the P reference is calculated by addition of accelerometer and centrifugal acceleration. So, we have:

$$A_{\mathrm{Cen}} = \omega_{\mathrm{Gyro}} \times V \qquad (7)$$

$$g_{\mathrm{ref}} = \mathrm{Acc} + \omega_{\mathrm{gyro}} \times V$$

Where: $\mathrm{Acc} = \begin{bmatrix} \mathrm{Acc}_x \\ \mathrm{Acc}_y \\ \mathrm{Acc}_z \end{bmatrix}$; $V = \begin{bmatrix} \mathrm{Velocity} \\ 0 \\ 0 \end{bmatrix}$

In addition to the reference measurement of the gravitational force, we need an estimation based on the Z - row of the direction cosine matrix. It is the projection of the G reference down-axis along the axes of the P reference. The roll-pitch rotational correction vector in P reference ($\mathrm{RP}_{\mathrm{Cor}}^{\mathrm{Plane}}$) is calculated by taking the cross product of the Z-row of R with the normalized gravity reference vector g_{ref} in formula (7), as follows:

$$\mathrm{RP}_{\mathrm{Cor}}^{\mathrm{Plane}} = \begin{bmatrix} -\sin\theta \\ \sin\varphi\cos\theta \\ \cos\varphi\cos\theta \end{bmatrix} \times g_{\mathrm{ref}} \qquad (8)$$

The calculation adjustment process to ensure the conditions of the orthogonal vectors, with the symbol as $\forall$, is called "Normalization", with the symbol as $\sim$, following three steps. Firstly, we compute the dot product of X-row and Y-row of R, which is supposed to be zero. The result is a measurement of how much X-row and Y-row are rotating toward each other:

$$X = \begin{bmatrix} r_{xx} \\ r_{xy} \\ r_{xz} \end{bmatrix};\ Y = \begin{bmatrix} r_{yx} \\ r_{yy} \\ r_{yz} \end{bmatrix};\ Error = X \bullet Y = X^T Y \tag{9}$$

Secondly, we split the errors for each vector so that the remaining bias after adjusting process will be lower than downright errors for just one vector. Apportioning half of the errors for each of X-row and Y-row, and rotating approximately X-row and Y-row in the opposite direction by cross coupling we have X-orthogonal and Y-orthogonal, and by taking cross product of X-orthogonal and Y-orthogonal, we have Z-orthogonal shown as below:

$$X_{\forall} = \begin{bmatrix} r_{xx} \\ r_{xy} \\ r_{xz} \end{bmatrix}_{\forall} = X - \frac{Error}{2} Y;\ Y_{\forall} = \begin{bmatrix} r_{yx} \\ r_{yy} \\ r_{yz} \end{bmatrix}_{\forall} = Y - \frac{Error}{2} X\ \ Z_{\forall} = \begin{bmatrix} r_{zx} \\ r_{zy} \\ r_{zz} \end{bmatrix}_{\forall} = X_{\forall} \times Y_{\forall} \tag{10}$$

The last step in the normalization process is to scale three rows of R to ensure that each of them has a magnitude equal 1. By using a Taylor's expression, we have the resulting magnitude adjustment equations for each row vector and have formula:

$$\begin{aligned} X_{\sim} &= \frac{1}{2}(3 - X_{\forall} \cdot X_{\forall})X_{\forall} \\ Y_{\sim} &= \frac{1}{2}(3 - Y_{\forall} \cdot Y_{\forall})Y_{\forall} \\ Z_{\sim} &= \frac{1}{2}(3 - Z_{\forall} \cdot Z_{\forall})Z_{\forall} \end{aligned} \tag{11}$$

Correction of yaw and roll-pitch is adjusted depending on COG and on acceleration weight (w). Using level arm correction by the constraint function, we calculate vector scale and vector addition to put data to PI to have total correction (T_{Cor}):

$$\begin{aligned} T_{Cor} &= W_{RP} \cdot RP_{Cor}^{Plane} + W_Y \cdot Yaw_{Cor}^{Plane} \\ \omega_{PCor} &= K_P \cdot T_{Cor} \\ \omega_{ICor} &= \omega_{ICor} + K_I dt \cdot T_{Cor} \\ \omega_{Cor} &= \omega_{PCor} + \omega_{ICor} \end{aligned} \tag{12}$$

Calculating the updated rotation matrix elements following every step of Omega_Vector, where ω_{CorGyr} is the corresponding element of gyro-x-roll, gyro-y-pitch, gyro-z-yaw of Corrected Gyro_Vector of data from equation (11).

$$\begin{aligned} R_{0,0}^U &= 0 & R_{1,2}^U &= -T_{INS} \times \omega_{CorGyr0} \\ R_{0,1}^U &= -T_{INS} \times \omega_{CorGyr2} & R_{2,0}^U &= -T_{INS} \times \omega_{CorGyr1} \\ R_{0,2}^U &= T_{INS} \times \omega_{CorGyr1} & R_{2,1}^U &= T_{INS} \times \omega_{CorGyr0} \\ R_{1,0}^U &= T_{INS} \times \omega_{CorGyr2} & R_{2,2}^U &= 0 \\ R_{1,1}^U &= 0 \end{aligned} \tag{13}$$

The Euler angles will be calculated according to new updated values of the matrix rotation element $\mathrm{R}^{\mathrm{U}}_{\mathrm{i,j}}$ from (13).

$$\begin{aligned}
\mathrm{pitch} &= -\mathrm{asin}(\mathrm{R}^{\mathrm{U}_{\sim}}_{2,0}) \\
\mathrm{roll} &= \mathrm{atan2}(\mathrm{R}^{\mathrm{U}_{\sim}}_{2,1}, \mathrm{R}^{\mathrm{U}_{\sim}}_{2,2}) \qquad (14) \\
\mathrm{yaw} &= \mathrm{atan2}(\mathrm{R}^{\mathrm{U}_{\sim}}_{1,0}, \mathrm{R}^{\mathrm{U}_{\sim}}_{0,0})
\end{aligned}$$

Adjustment of the output values of the gyro vector by adding the compensation values for the raw signal and then putting back into the rotation R update equation to re-calculate the elements of R. At this point, the program has completed one full cycle of calculations. Next steps we repeat the entire process to complete the DCM algorithm, in which the parameters of PI can be changeable depended on error of tilt angles and *w* processing as follow:

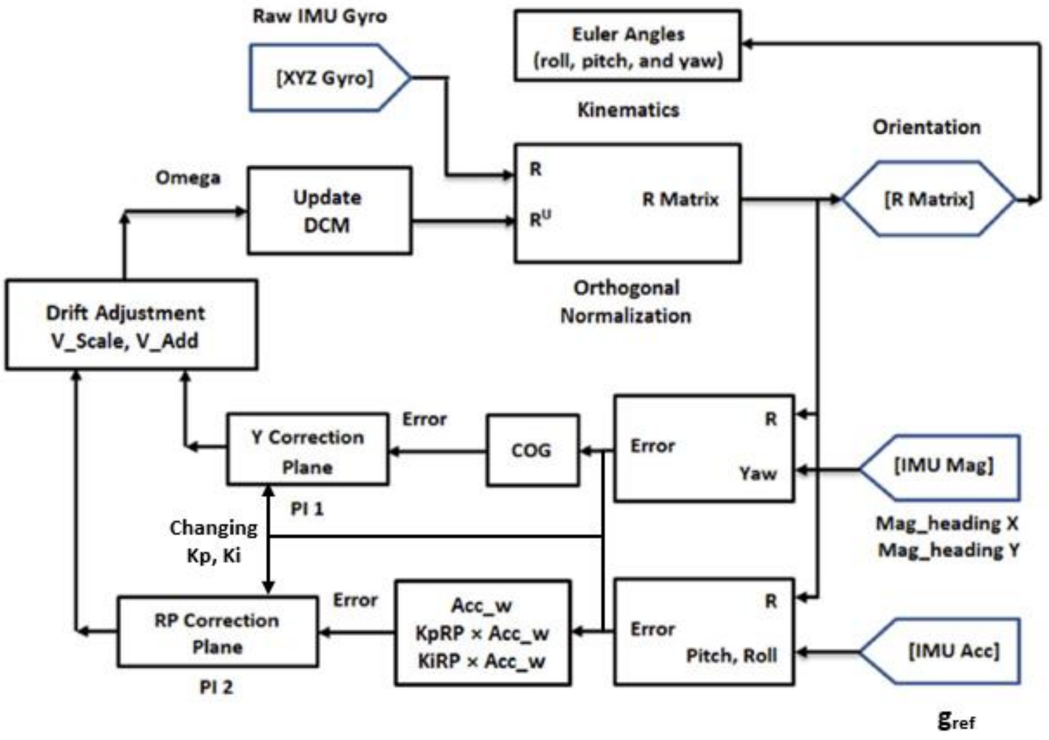


***Figure 1**. The diagram of developed DCM algorithm*

Surveying of a MEMS INS 9-DOF GY85 in static mode in both cases: not using DCM (Figure 2.A) and using DCM (Figure 2.B), we have measurement data of the sensor acc, Mag, Gyro given a small fluctuation. In fact, the values of the IMUs (Inertial Measurement Unit), including Acc ADXL345, Gyro ITG-3200, and compass HMC5883L, do not depend on whether or not using the DCM algorithm. The difference between two cases of surveying is apparent when using these base signals to calculate roll, pitch and yaw angles, shown as below:

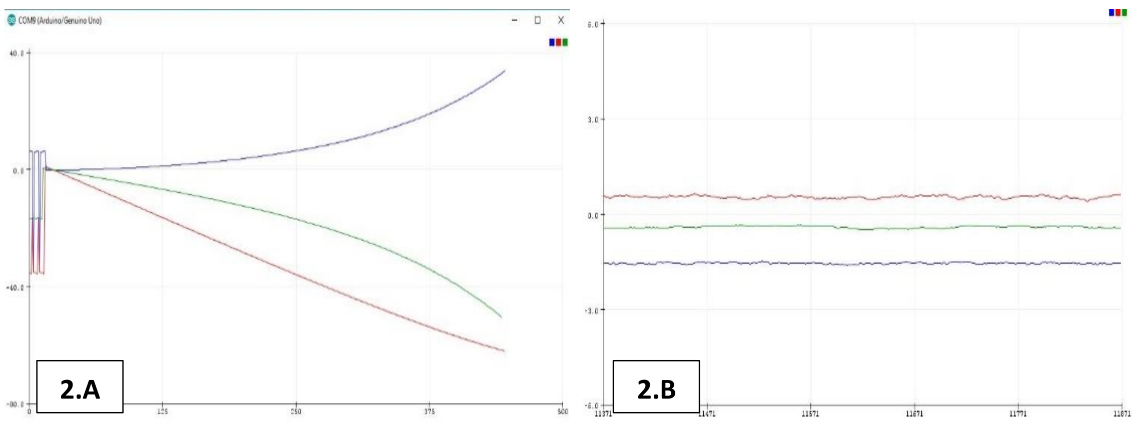


***Figure 2**. Roll, pitch and yaw angles of GY85 in case of not using (2.A) and using (2.B) the developed DCM algorithm*

In case of not using updated parameters of DCM in the algorithm, the signals of roll, pitch and yaw were drifting rapidly. The data are not stable, increase/decrease continuously over time. With GY85, at the interval of 2 minutes, deviation of roll is 50^0, pitch is 30^0 and yaw varies up to the 120^0. This achievement led to big accumulating errors when calculating the rotation matrix element $r_{i,j}$, thereby causing major discrepancies in the velocity and position. When applying updated parameters of DCM in the algorithm for this GY85 in static mode, the values of roll, pitch and yaw are not varied for long working time. During more than 2 minutes and longer, they are still given out at nearly the same results. The maximum changing values below 0.4^0 for roll, 0.6^0 for pitch, and 0.6^0 for yaw. These values are not cumulative over time and automatically corrected. The measured values indicate they oscillate around equilibrium and stability during the working time of the INS.

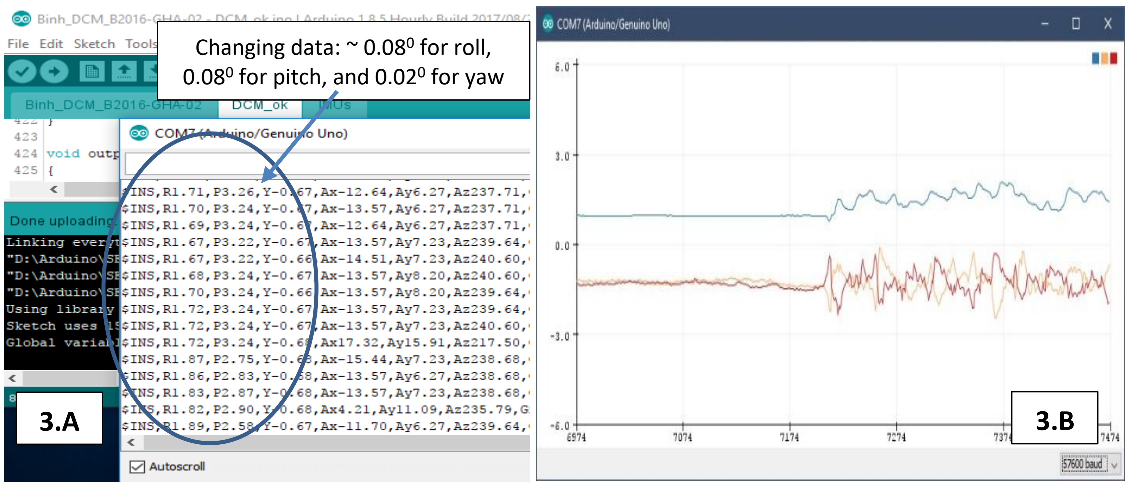
III. APPLICATION IN BALANCING CONTROL DRONES

After calibrating INS to have ranges of IMU accelerator, gyro and magnetometer, we finished testing INS and had good results with developed DCM algorithm described above, then we designed and created a drone as H-Quadcopter type to check balance control, including: 4X Racerstar Racing Edition BR2212 980KV 2-4S Brushless Motor, Gemfan 1045 Carbon Nylon CW/CCW Propeller, SimonK 30A 3S Brushless ESC with 5V 3A ESC for RC Model, ZOP Power 22.2V 5500 mAh 6S 60C Lipo Battery, Matek Mini Hub Power Distribution Board with BEC 5V and 12V for FPV Multicopter, and FlySky FS-T6 2.4 GHz Digital Proportional 6 Channel Transmitter-Receiver System. The experimental design in this research used common devices such as Arduino Uno and an INS 9-DOF GY85 to check algorithm. There were some devices integrated on drone as GPS module, RF module, Camera and some other devices, as shown in figure 4.A.

Operating GY85 as setting of acc is ±8g, the output value of acc will be 4096 when IMU in rest. Real acc values are nearly reserved values +4134 in normal position, and – 4120 when acc flips upside down. When IMU is at angle of 45^0, the output values change from 4134 to 2922.

The output value of sensitivity scale factor for gyro is selected of 65.5, when gyro value is changed 1^0/s. Angular velocity is 1^0/s, that means Z-axis_out equal 65.5. The gyro values are needed to be set to zero by subtracting from output. Data output refresh cycle is 250 Hz, so error for one rotation cycle is 0.1845^0. In cycle of one minute, the value I get is: 393 (^{0}/s) * 60s = 23580. It means: 23580/65.6 = 360^0. Data output refresh cycle is 250 Hz (1/4 ms).

When accelerating by shacking IMU in different direction, these angles are changed very fast, gone all over and become completely useless. This is the same thing what happened on the drone because of vibration presented when motors are spinning. That will make values unreliable. When filtered these data, they become more useful. That means the same thing when we have less vibration by using filter to solve raw IMU data and this DCM algorithm. The vibration result was very good when spinning four motors to keep the drone flying stable on the sky as in figure 4.B, and had results described in figure 3.B as below. Maximum yaw, roll, pitch angles were very small as around maximum of 1^0, 1.3^0 and 1.3^0, varied a little bit and stable when spinning motors changed for testing balance of drone in other altitude as in flying action.


Binh_DCM_B2016-GHA-02 - DCM_ok.ino | Arduino 1.8.5 Hourly Build 2017/08/
File Edit Sketch Tools
Binh_DCM_B2016-GHA-02
DCM_ok
IMUs
Changing data: ~ 0.08^0 for roll, 0.08^0 for pitch, and 0.02^0 for yaw
COM7 (Arduino/Genuino Uno)
Done uploading
Autoscroll
3.A
COM7 (Arduino/Genuino Uno)
6.0
3.0
0.0
-3.0
-6.0
6974
7074
7174
7274
7374
7474
57600 baud
3.B

```
422 }
423
424 void outp
425 {
```

```
Linking everyt
"D:\Arduino\SE
"D:\Arduino\SE
"D:\Arduino\SE
Using library
Sketch uses 15
Global variabl
```

```
$INS,R1.71,P3.26,Y-0.67,Ax-12.64,Ay6.27,Az237.71,
$INS,R1.70,P3.24,Y-0.67,Ax-13.57,Ay6.27,Az237.71,
$INS,R1.69,P3.24,Y-0.67,Ax-12.64,Ay6.27,Az237.71,
$INS,R1.67,P3.22,Y-0.67,Ax-13.57,Ay7.23,Az239.64,
$INS,R1.67,P3.22,Y-0.66,Ax-14.51,Ay7.23,Az240.60,
$INS,R1.68,P3.24,Y-0.67,Ax-13.57,Ay8.20,Az240.60,
$INS,R1.70,P3.24,Y-0.66,Ax-13.57,Ay8.20,Az239.64,
$INS,R1.72,P3.24,Y-0.67,Ax-13.57,Ay7.23,Az239.64,
$INS,R1.72,P3.24,Y-0.67,Ax-13.57,Ay7.23,Az240.60,
$INS,R1.72,P3.24,Y-0.68,Ax17.32,Ay15.91,Az217.50,
$INS,R1.87,P2.75,Y-0.68,Ax-15.44,Ay7.23,Az238.68,
$INS,R1.86,P2.83,Y-0.68,Ax-13.57,Ay6.27,Az238.68,
$INS,R1.83,P2.87,Y-0.68,Ax-13.57,Ay7.23,Az238.68,
$INS,R1.82,P2.90,Y-0.68,Ax4.21,Ay11.09,Az235.79,G
$INS,R1.89,P2.58,Y-0.67,Ax-11.70,Ay6.27,Az239.64,
```

Figure 3. *Roll, pitch and yaw angles of testing vibration data of drone in static mode (3.A) and dynamic mode of spinning motors (3.B).*

The IMU gyro is sensitive to vibrations, drifts a little over time. Very fast vibration also makes the raw data of IMU accelerometer to be useless. When the IMU started, and after calibration of gyro, raw angles are needed to be set by the accelerometer. It is just possible when accelerometer isn't rest. One more problem occurs when flying far away and having long turn in high velocity. In this far away turning in flying situation, the acceleration values of drone are changed by generating a small shake because of gravitational vector to be angled instead of pointing straight down. All these problems were solved by DCM algorithm described above and helped drone very good at balancing with small vibration, as shown in figure 3.B and figure 4.B. Finally, the drone can fly well, both in manual mode and automatic mode, as shown in figure 4.C.

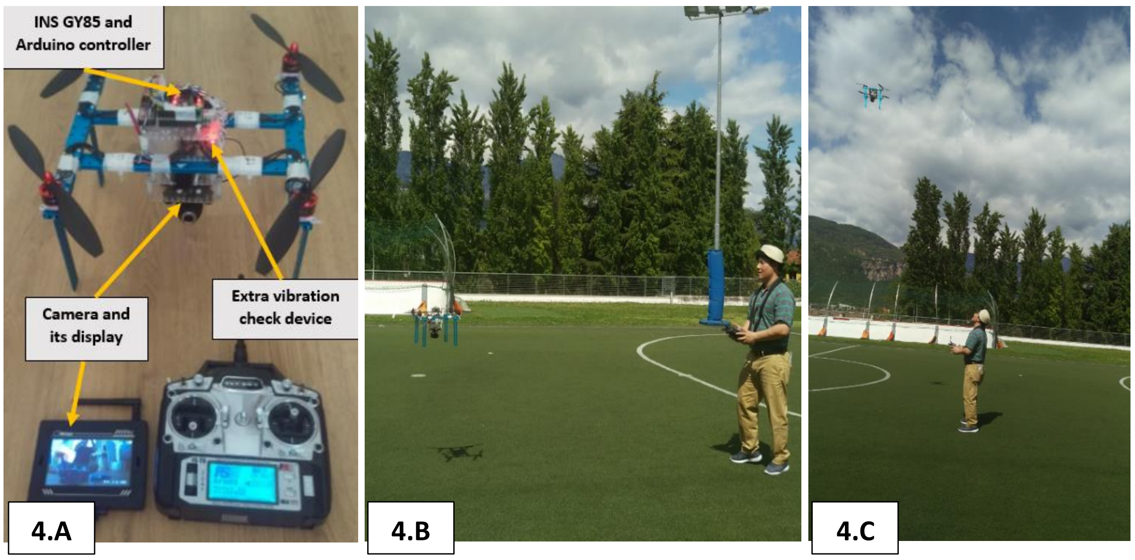


***Figure 4**. The drone, testing balance and flying action.*

III. CONCLUSION

The new feature in this research is an autonomous anti-drift R method based on DCM solution using MEMS INS 9-DOF developed for autonomous devices. Our solution in tilt angle processed from internal Mag, distributed error and updated elements of R form flexible PI had solved accumulated errors in INS and may help MEMS INS 9-DOF systems to operate independently.

Most of the gyro offset varying with supply voltage can be removed by measuring the offset during power up, which provided the IMU capability of keeping motionless at that time. In fact, the variation of offset caused by supply voltage and temperature is usually rather slow, so that DCM method can continually remove the offset and maintain balancing drone easily. Drift around all three axes can be completely eliminated as long as the drone moves continuously. When the drone stopped moving, yaw drift occurred. These values depended on the residual offsets of the gyro, and were almost removed completely by DCM solution developed above. When motors of drone start moving, the drift will be cancelled automatically. This is fundamental to develop and apply in specialized flight controllers as Pixhawk or ArduPilot APM to have better shots and videos. This makes image processing easier in the next research step.

Acknowledgements

This research was funded by the Ministry of Education and Training (MoET) under grant number B2016-GHA-02

References

[1]. N. H. Q. Phuong, H.-J. Kang, Y.-S. Suh, and Y.-S. Ro, “A DCM based orientation estimation algorithm with an inertial measurement unit and a magnetic compass”. J-JUCS - Journal of Universal Computer Science, 2009, vol. 15, no. 4, pp. 859–876.

[2]. Pengfei Guo, Haitao Qiu, Yunchun Yang, Zhang Ren, “The soft iron and hard iron calibration

method using extended Kalman filter for attitude and heading reference system". Position, Location and Navigation Symposium, USA, 2008.
[3]. W. Premerlani and S. Balutal, "DCM tutorial - an introduction to orientation kinematics", Stalino Electronics, USA 2009.
[4]. G. Baldwin, R. Mahony, J. Trumpf, T. Hamel, and T. Cheviron, "Complementary filter design on the special euclidean group SE(3)". European Control Conference (ECC), 2007, pp. 3763–3770.
[5]. R. Mahony, T. Hamel, and J. M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group". IEEE Transactions on Automatic Control, vol. 53, no. 5, pp. 1203–1218, 2008.
[6]. J. Hartikainen, A. Solin, and S. Sarkka, "Optimal filtering with Kalman filters and smoothers - A manual for the matlab toolbox ekf-ukf," GNU, FINLAND, 2011♦

MODEL-BASED TESTING FOR SAFETY-CRITICAL …

(Following page 83)

ACKNOWLEDGEMENT

This research was supported by National Natural Science Foundation of China (Grant No. 61703349), Key Research Projects of China Railway Corporation (Grant No. 2017X007-D), Fundamental Research Funds for the Central Universities (Grant No. 2682017CX101, 2682017ZDPY10) and Opening Foundation of Gansu Provincial Key Laboratory of Traffic Information Engineering and Control (Grant No. 20161103).

References
[1]. C. Rao, J. Guo, N. Li, Y. Lei, Y. Zhang, Y. Li, and Y. Cao, "Applying combinatorial testing to high-speed railway track circuit receiver," in 2017 6th Internaltional Workshop on Combinatorial Testing (IWCT2017),Tokyo, Japan, March 2017, pp. 199–207.
[2]. A. Petrenko, A. Simao, and J.C Maldonado, "Model-based testing of software and systems: recent advances and challenges," International Journal on Software Tools for Technology Transfer, no.14, pp.383-386, June 2012.
[3]. T.S Chow, "Testing design modeled by finite-state machines," IEEE Transactions on Software Engineering, vol.4, no.3, pp.178-187, May 1978.
[4]. S.H. Li, J. Wang, Y.C. Qi, "Model-Based Methods for Real Time System Testing," COMPUTER ENGINEERING & SCIENCE, vol.28, no.4, pp.119-123, 2006
[5]. J. Yang, J. Wang, H.W Chen, "A review of Model-Based for Software Testing," COMPUTER SCIENCE, vol.31, no.2, pp.184-187, 2004
[6]. C F Hu, J Guo, N Li, et al. Towards Effective and Scalable Testing for Complex High-Speed Railway Signal Software[C]// IEEE International Conference on Software Quality, Reliability and Security Companion. IEEE, 2017:571-572.
[7]. N. Li, A. Escalona, and T. Kamal, "Skyfire: Model-based testing with cucumber," in 2016 IEEE International Conference on Software Testing, Verification and Validation (ICST), April 2016, pp. 393–400.