Experiments of low-cost INS/GPS Navigation platform based on PC104

N. Hachelef, P. Lavoie, D. Li, R. Landry
Department of Electrical Engineering
École de Technologie Supérieure (ETS), LACIME Laboratory
1100 Notre Dame Street West, H3C 1K3 Montreal, Canada
nacim.hachelef.1@ens.etsmtl.ca philippe.lavoie.5@ens.etsmtl.ca rlandry@ele.etsmtl.ca di.li@etsmtl.ca

Biography

Nacim Hachelef received a BS degree in engineering techniques at Versailles University (France). He is a Master Student in Aerospace specialized in control, at ESTACA University, Paris (France). He was an exchange internship student at École de Technologie Supérieure, Montreal (Canada) on GPS/INS integration. His research activities involve in MEMS and RLG-based INS/GPS integration in simulation/real time tests.

Abstract

With the state of the art innovations of navigation systems, GPS is becoming a very important part of daily life. Indeed, GPS is a very helpful and precise navigation system however its performance is fairly vulnerable due to the environment noise, which may cause GPS signals loss and attenuation. Hence this is hard to rely much on this technology.

Compared with GPS, the INS is a self-contained navigation system, which has jamproof performances but the sensors induce errors which are growing with time.

These two navigation systems integrated together can compensate each other's weakness providing a continuous data acquisition using an adapted filter. This integrated solution of GPS/INS gives an accurate solution which is low-cost and jamproof.

This paper presents a low-cost GPS/INS platform developed with Micro Electro-Mechanical System (MEMS). Fast prototyping tools such as Matlab/Simulink are utilized to implement/validate the algorithm. This platform is consist of an assembly of different PC104 cards such as acquisition card, GPS card, Ethernet card, mother board… stacked up together providing the link between the sensors, the algorithm and the host computer.

The software including INS, GPS data acquisition and the Kalman filter is loaded and processed on the PC104 based computer. This platform is totally self-embeddable and can be used independently without any external hardware/software supports. The algorithm utilized can be adapted to different grade of inertial sensors by changing few parameters.

1. Introduction

High precision Inertial Navigation employing the expensive inertial sensors is contained to high technology applications like satellite, probe, missiles… but the emergence of low-cost MEMS in the market has facilitated the development of new applications. Therefore, inertial systems are expanding its market to more common applications such as car navigation, video games and robots.

If combined with GPS, inertial navigation system can be a very precise device delivering data even when the GPS signals are blocked. During the GPS loss, inertial sensors such as MEMS can provide position, velocity and acceleration until the GPS can find a new available satellite constellation. In that case, the precision is not the finest but with the Kalman filter running, the drift caused by the sensors errors can be compensated when the GPS signals are found.

The MEMS platform is totally embeddable providing a totally autonomous system when powered with an embeddable power supply. The platform utilized is based on a PC104 bus computer by which we can integrate all the needed hardware devices.

The paper is organized as the followings. The section two gives a global view of the platform: the devices utilized, the software used to communicate and the algorithm structure. The section three shows the MEMS test results including simulation and real time acquisition followed by a qualitative analysis. Section four shows the results by utilising the same algorithm with different sensors which demonstrates the versatility of the algorithm. Finally, the results are shown in the paper and the future advancement of the project are also discussed in the conclusion.

2. Hardware and Software

A good link between the hardware and the software is a key issue for the design of the platform. This link is provided by the xPcTarget interface (Simulink) which gives a correct and quick link for the communication between the host computer and the PC104.

2.1 Hardware

The platform is composed of seven different stages stacked up together through the PC104 bus. This configuration makes the individual element independent from each others and provides a good synchronization for the data transmission.

On the top stage there is the inertial sensor (MEMS) which is not PC104 compatible but the information is sent via an external bus to the acquisition card. This stage provides the specific forces and angular rates measurements given by the MEMS.

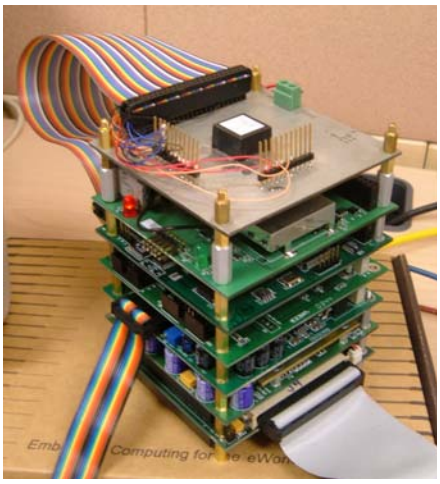


Figure 1: GPS/INS Platform Prototype

On the next stage, there is the GPS receiver which obtains and transmits the position and velocity solutions to the algorithm via the PC104 bus.

Following the GPS receiver, there is a 16-bit-acquisition card which collects the data from the MEMS via the external bus.

Under that, a PC104 compatible power card delivers the power and voltage required to each other cards.

Then, a PC104 compatible network card is added to build a TCP/IP connection to a host computer. This connection is used for two important tasks: first, to send the programs for the real time processing and second, to acquire and record the data processed in real time.

This platform is also equipped with a graphic card making able to display all the data processed by the PC104 in real time on a distinct screen.

On the bottom stage, there is the PC104 motherboard where the algorithm is loaded and executed.

The benefit provided by PC104 compatible devices is that the data are already synchronized because of the unique PC104 bus linking all the devices.

2.2 Software

There are two distinct procedures detailed in the following section, one for the simulation and one for the real time acquisition. The equations utilized by the software are implemented into the different blocks of the shown on the figure 2:

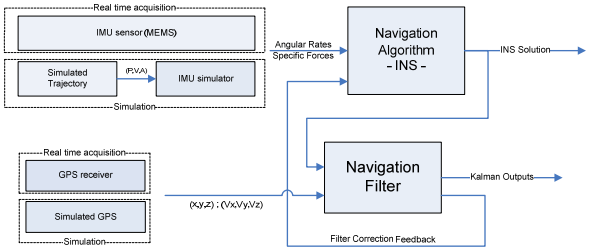


Figure 2: Program Structure: INS and Kalman filter

For the simulation, the reference trajectories are generated by the software Microsoft Flight Simulator®. The information provided by the software can be saved directly in an ASCII format file which is compatible to Matlab/Simulink. The useful data provided by the simulator are the following:

- Position expressed in terms of 'latitude' [deg], 'longitude' [deg] and 'altitude' [ft]

- Velocity expressed in terms of 'North Velocity', 'East Velocity' and 'Vertical Velocity', all in [ft/s]

- Attitude angles expressed in terms of 'pitch' [deg], 'roll' [deg] and 'heading' [deg]

The IMU simulator is a program designed to generate the angular rates and specific forces for a simulated flight trajectory by using the already-known position, velocity and attitude angles. The IMU data generation takes into account the gravity and the Earth's rotation. In general, for the sake of a better understanding this program is like a reverse procedure of the INS calculation which provides the position, velocity and attitude angles from the sensors measurements. To avoid redundancy, only the INS process will be explained.

To have a realistic simulation of the MEMS, the noises and bias for the three axes of the accelerometers and gyroscopes have to be taken in account. This equation shows how those parameters are added to the raw measurement in the algorithm:

$$\vec{X}_{mes} = (I + M_s) \times (\vec{X}_{true} + \vec{\delta}x + \vec{\varepsilon}) \qquad [5]$$

Where:

$\vec{X}_{mes}$ = measured variable of the sensor,
[m/s^2] for accelerometer and [rad/s] for gyroscope
$\vec{X}_{true}$ = noise free variable (same units as X_{mes})
M_S = scaling factor matrix
$\vec{\delta}x$ = bias on the sensor measurement
$\vec{\varepsilon}$ = noise of the sensor measurement
I = identity matrix

The corresponding Simulink model for the introduction of the perturbations is shown on the figure 3:

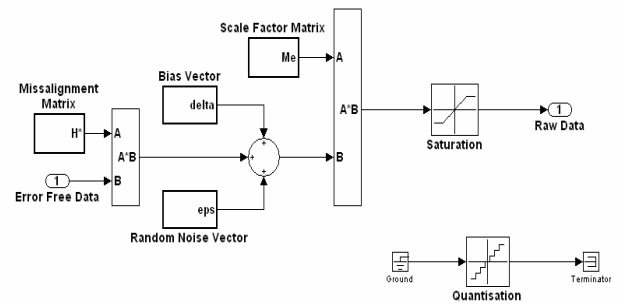


Figure 3 : Accelerometers and Gyroscopes model

In this model the bias and the noise vectors are added to the error free data. Then, the matrix is multiplied by a scale factor to obtain the simulated raw data. The values for those parameters are provided by the manufacture specification datasheet.

This IMU model takes the main characteristics of the MEMS in consideration. However, it still can be improved with other characteristics such as influence of temperature and non linearity as long as the information is provided by the manufacturer.

The rest of the program is the same for both simulation and real time acquisition. The raw data coming from the MEMS or simulated with the IMU simulator are processed directly by the INS algorithm which performs all the calculations to obtain the position, velocity and attitude angles. The following equations are used in the model and they are integrated with a method using quaternion:

$$\left[{}^{E}\dot{\boldsymbol{v}}_N \right]_N = {}^{N}C_B[{}^{I}\boldsymbol{a}_B]_B + [\boldsymbol{g}_p]_N - \left(S\{[{}^{E}\boldsymbol{\omega}_N]_N\} + 2S\{[{}^{I}\boldsymbol{\omega}_E]_N\}\right)\left[{}^{E}\boldsymbol{v}_N\right]_N \qquad (1)$$

$${}^{N}\dot{C}_B = {}^{N}C_B S\left\{\left[{}^{I}\boldsymbol{\omega}_B\right]_B\right\} - S\left\{\left[{}^{I}\boldsymbol{\omega}_N\right]_N\right\}{}^{N}C_B \qquad (2)$$

$${}^{E}\dot{C}_N = {}^{E}C_N S\left\{\left[{}^{E}\boldsymbol{\omega}_N\right]_N\right\} \qquad (3)$$

$$\dot{h} = \left[{}^{Z}\boldsymbol{u}_N\right]_N * \left[{}^{E}\boldsymbol{v}_N\right]_N \qquad (4)$$

Where:

$[{}^{I}\boldsymbol{a}_B]_B$ is the measured acceleration vector;
$[\boldsymbol{g}_p]_N$ is the gravity vector at the surface of the earth;
$[{}^{E}\boldsymbol{\omega}_N]_N$ is the transport rate $\boldsymbol{\rho}^N$;
$[{}^{I}\boldsymbol{\omega}_B]_B$ is the measured angular rate vector;
$\left[{}^{Z}\boldsymbol{u}_N\right]_N$ is the vertical unit vector of the Navigation
I is the inertial frame;
${}^{N}C_B$ is the transfer matrix from the Body frame to the Navigation frame;
${}^{E}C_N$ is the transfer matrix from the Navigation frame to the Earth frame;
E is the Earth Centered Earth fixed (ECEF) frame;
N is the Navigation frame;
B is the Body frame (vehicle-fixed frame);
S{a} is the anti-symmetric matrix of the vector a.

The first equation (1) shows how to get the velocity propagation of the navigation frame in the navigation frame (N) referred to the Earth frame (E) from the IMU output and the gravitation model.

The second equation (2) explains the propagation of the Body frame (B) compared with the Navigation frame (N).

The third equation (3) is the propagation of the position matrix.

The fourth equation (4) represents the propagation of the altitude.

The whole equation allows the calculations of PVA from the IMU raw measurements independently. This means that no correction to the measurements have been done yet.

The gravity model takes in consideration of the Earth's mass and Earth's rotation:

$$ {}^{I}\vec{g}_{pN} = {}^{I}\vec{g}_{N} + S({}^{I}\vec{\omega}_{E}).S({}^{I}\vec{\omega}_{E}).{}^{I}\vec{r}_{N} $$

Where:

${}^{I}\vec{g}_{pN}$ is the Plum-bob gravity vector

${}^{I}\vec{g}_{N}$ is the gravity vector

$S({}^{I}\vec{\omega}_{E})$ is the Earth's rotation rate

${}^{I}\vec{r}_{N}$ is the distance vector from center of the Earth to the mobile

The Earth's rotating rate is also taken into account. For this reason, when the sensors are static, they can measure the Earth's rotating rate: $\omega_E \approx 15.04$deg/hour.

Concerning the attitude angles the equation is the following:

$$ \omega_{IB}^{B} = \omega_{IE}^{B} + \omega_{EN}^{B} + \omega_{NB}^{B} \qquad (5) $$

Where:

ω_{IB}^{B} is the gyroscopes measured value

ω_{IE}^{B} is the Earth's rotating rate

ω_{EN}^{B} is the transport rate i.e. the rotation rate of the Navigation frame with respect to the Earth's frame

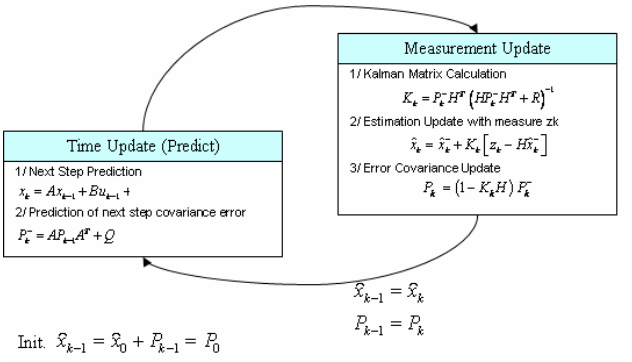
ω_{NB}^{B} is the attitude rate: rotation rate of the Navigation frame with respect to the Body frame

The equation (5) shows the composition of the rotation vector for Body frames. This is the process to obtain the **gyroscopes measures by using the attitude rates, the** transport rate and the Earth's rotating rate.

When the GPS signals are available, corrections provided by the Navigation filter are taken into account during the INS process. Currently the role of the Navigation Filter is played by a Kalman Filter that provides the necessary corrections to input into the INS.

The role of the Kalman filter in our application is to estimate and correct the IMU measurements (position, velocity and attitude angles) with the aid of the GPS position and velocity. The Kalman filter uses the estimated state from the previous time step and the current measurements to compute the estimated state of the current time step.

This is a general representation of the Kalman filter cycle which content two important parts, the prediction and the measurement update:



Where:

$\hat{x}_k$ is the estimate of the state at time k;

$P_{k,k}$ is the error covariance matrix (a measure of the estimated accuracy of the state estimate).

The Navigation filter and the INS algorithm are continuously being modified and improved. Therefore, it is essential to be able to change some parts of the algorithm rapidly. From this point of view, Simulink is a very convenient tool for fast prototyping. The algorithm can be easily modified and with the *real time workshop* tool, the whole algorithm can be re-make in C, complied and then loaded on the platform. The *xPcTarget* library of Simulink provides the necessary drivers to read directly the data from the different cards. This library can create the interface with the host computer to read and command the platform in real time.

While from a different point of view, Simulink is not the ideal software to run this algorithm due to all the operation needed to pass from the Simulink version to the C version loaded into the platform. This sequence of operation can cause some unexpected problems decreasing the performances of the algorithm. Therefore, a C-code version of the whole algorithm package is currently under development..

3. MEMS Results

The following section will present the test that has been made for both simulation and real time acquisition with the MEMS sensors.

3.1 Simulation

The Figure 4 represents the trajectory processed in simulation. The blue line is the reference trajectory generated by the flight simulator and the red line corresponds to the solution calculated by the algorithm.

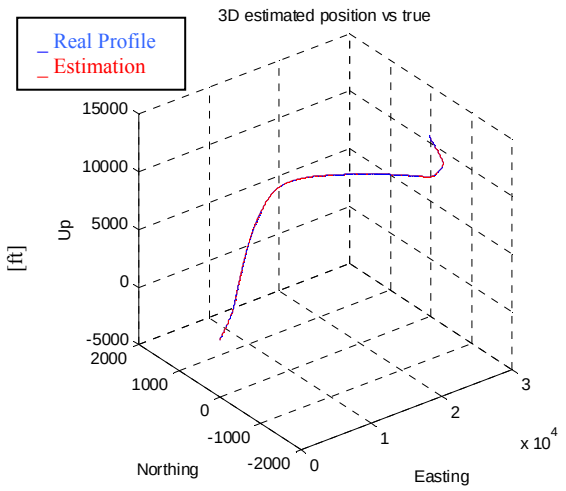


Figure 4 : 3D Position of MEMS simulated sensors

Position: The Kalman filter update rate can be observed clearly on the graph: every 10 ms, the Navigation filter provides a new estimation of the position errors. The maximal error that is located on an axis is around 5 meters. This period correspond to the high dynamic period during the flight.

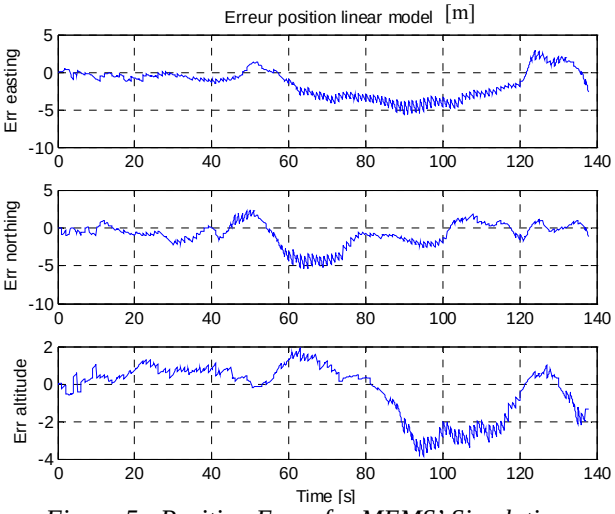


Figure 5 : Position Error for MEMS' Simulation

P matrix: This matrix gives the information about the Kalman filter estimations performance. Its convergence means that the estimations are close to the reality. The P matrix of this experiment shows that all the parameters estimations are converging quickly. This proves that the kalman filter is working well giving a good estimation of the parameters.

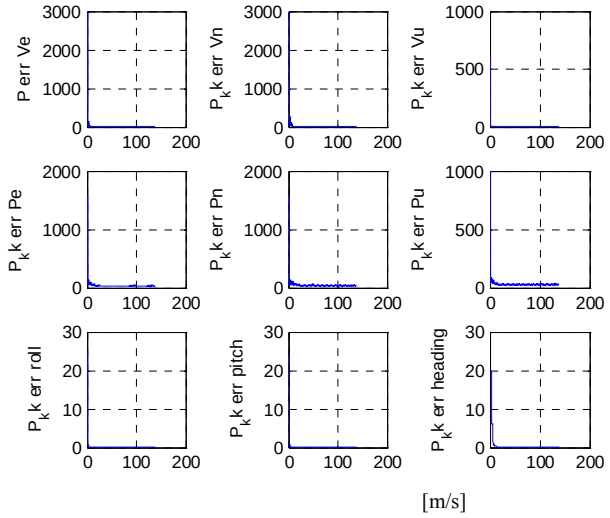


Figure 6 : P Matrix for MEMS' Simulation

Velocity: The results show that the error from the low-cost MEMS (Simulation) never exceeds one percent of the total value. Considering the MEMS characteristics, the velocity errors shown are good.

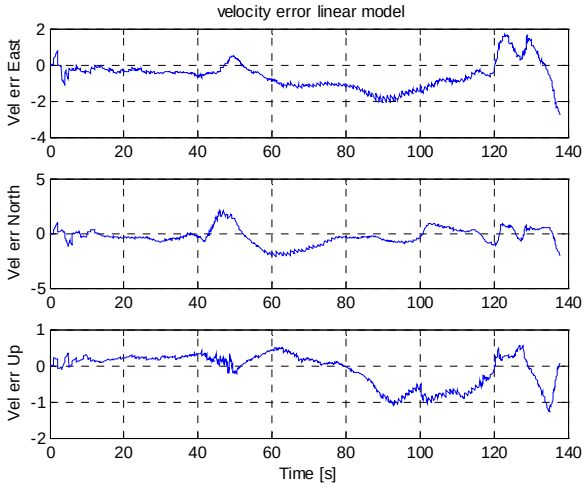


Figure 7 : Velocity Error for MEMS' Simulation

The experimental results validate the algorithm design. This algorithm converges quickly and the final results provide a good estimation of the initial trajectory.
It is also important to notify the limitations of the model. In this situation, the noise, biases and all the major errors are added manually. That means that the values of the added noises are known exactly however which is impossible in real time world.

Another crucial aspect is that all those parameters are dependent of many different issues such as temperature, humidity, pressure, sensor non-linearity etc. Usually, an estimation of the action of the main parameters is provided by the datasheet but it is not always simple and convenient to replicate all those effects.

3.2 Real Time Experiment

The experiment shown in the this (?) section is a stand alone test that has been made with the low-cost sensors. The purpose of this test was to test the performances of the MEMS without any aiding devices such as GPS assistance. The trajectory is a very simple trajectory made inside a building which we know precisely the length. The figure 8 corresponds to this trajectory:

Figure 8 : Real time acquisition trajectory (MEMS)

After processing the algorithm, the results are as shown in the figure 9:

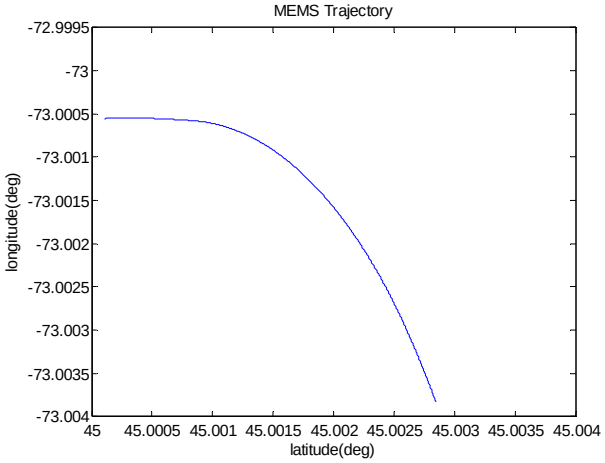


Figure 9 : Real time INS solution (MEMS)

The results coming from this experiment are provided in the table 1 [2].

	Real		MEMS
Total Distance	15.97m		8.30m
Total Error		48.02%	
Latitude Dist.	8.23m		5.32m
Longitude Dist.	13.67m		6.36m
Test Duration		20.36s	

Table 1 : Comparative Table

During these experiments it is possible to observe various characteristics of the sensor. The result shows that these sensors produce an error which is growing with time making it barely reliable for a long duration stand alone condition.

This experiment has been performed several times and it has been possible to show the repeatability of the results. This experiment also demonstrates that this device is completely self-contained. It works inside as well as outside always giving the same accuracy in the results.

Something important with MEMS is that it cannot set initial conditions itself. So if used alone, it has to be manually initialized to initial position, velocity and acceleration.

4. Other sensors

A lot of tests have been made with another type of sensors. The purpose for these tests was to validate the whole algorithm, to test its performances and to develop a new C-code version of it. The algorithm has been validated in both Simulink and C-code version with these sensors. As it was aforementioned the algorithm is versatile and the integration of the new sensors to the algorithm was made by changing just a few parameters which are mostly related to the sensors characteristics itself.

The results shown in the next section are related to a real time road test made in a parking lot. The dynamic of the trajectory is low and the total length is approximately five minutes. The figure 10 represents the whole trajectory:

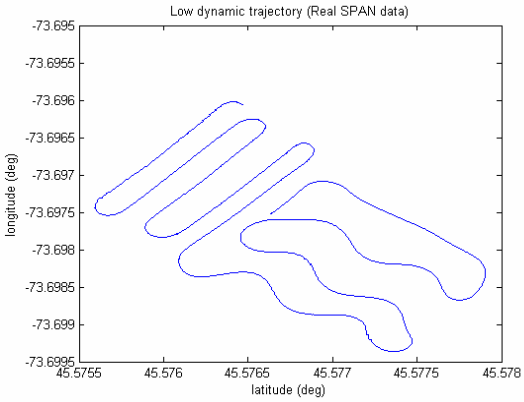


Figure 10 : Real time acquisition trajectory (other sensors)

For this test, the algorithm was processed with the C-code program which is faster and more accurate than the Simulink version. The solution calculated is the integrated GPS/INS solution and the results are the followings:

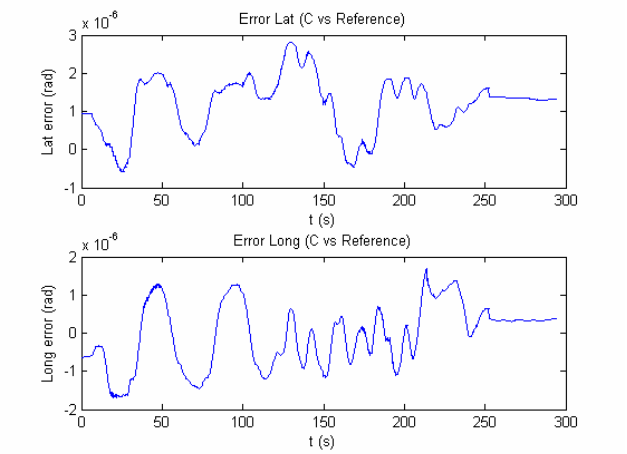


Figure 11 : Latitude and longitude errors (other sensors)

Position: The position error is represented in latitude and longitude error. The maximum errors found for this trajectory are around $\pm 2 \times 10^{-6}$ radians which correspond to approximately 0.5 to 1 meter.

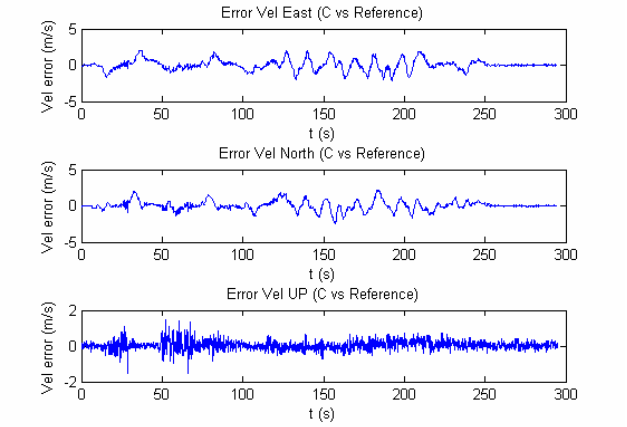


Figure 12 : Velocity errors (other sensors)

Velocity: The velocity errors are represented in meter per seconds. The maximum errors in this experiment are around 2 meter per second and seams to occur when the dynamic is high. These errors can be caused because of the low data acquisition frequency (limited to 100Hz for this sensor) so when the dynamic is high, some data are lost causing a lot of errors.

Attitude angles: The attitude angles errors are represented in radians. The maximum errors are around 0.002 radians (0.1°) for the roll and pitch and around 0.05 radians (1°) for the heading. The roll and pitch error are very small because of the nature of the trajectory. This test was made with a car on a plain road so the roll and pitch are not changing a lot during the whole trajectory.

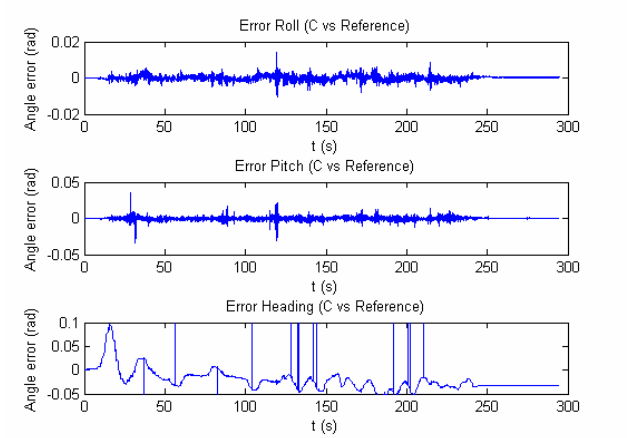


Figure 13 : Attitude errors (other sensors)

5. Conclusion

The results from the MEMS test shows that the position of the solution directly depends on the time duration of the simulation. This also explains the reason that such grade of sensor needs to be corrected with an aiding device, in this study, GPS plays this role.

Nevertheless, if this sensor is used for a limited time span, it can be helpful for its jamproof characteristic and it can still provide a signal when GPS is unavailable. In a few words, MEMS can be an alternative to GPS when the signals are blocked.

The results from this paper show that the algorithm is working well but there is still some work to do to improve it. For example, there is still no sensor compensation such as bias compensation which could reduce a lot of the drifting effect on the MEMS. Also, the MEMS do not provide the initial position, velocity and attitude angles so they have to be set up before running the algorithm. In a near future, an initial alignment process will be added to the algorithm making it able to initialize all these parameters by itself. The GPS signals used by the algorithm as an aiding device can be improved too with the integration of a RTK solution which is also under development.

The next goal for the project is to test the program with a different low-cost MEMS sensor on a real fly test to see the performance of the whole algorithm in a high dynamic environment.

References

[1] Giroux R. (August 2004), «Capteurs bas de gamme et système de navigation inertielle : Nouveaux paradigmes d'application», PhD Thesis, École de Technologie Supérieure, Montre, Canada.

[2] Giroux R., Landry R.Jr., Leach B. and Gourdeau R. (2003), «Validation and Performance Evaluation of a Simulink Inertial Navigation System Simulator», Canadian Aeronautics and Space Journal, Vol. 49, No. 4, p 149-161.

[3] Giroux R., Sukkarieh, S. and Bryson, M., «Implementation of a Skewed-Redundant Low-Cost INS in a Fast-Prototyping Environment», ION NTM 2004, San Diego, USA, 26 - 28 January 2004.

[4] Savage, P.G. (2000) «Strapdown Analytics, Part I», Strapdown Associates Inc, Maple Plain, Minnesota.

[5] Z. Bennour, R. Jr. Landry, R. Giroux, A. Constantinescu, G. Gavidia «Web-Based MEMS Inertial Navigation Simulator»

[6] Giroux, R., Gourdeau R. and Landry R.Jr., «Extended Kalman filter real-time implementation for low-cost INS/GPS Integration in a Fast-prototyping Environment», 16th Canadian Navigation Symposium, CASI , Toronto, Canada, 26 - 27 April 2005.

[7] MEMSense «www.memsense.com», AccelRate3d version datasheet