Timestamping for an array of low-cost sensors

E. Dorveaux * **D. Vissiere** ** **A.P. Martin** *** **N. Petit** ****

* *Mines-Paris Tech, Paris, France*
(e-mail: eric.dorveaux@mines-paristech.fr)
** *DGA, Vernon, France*
(e-mail: david.vissiere@dga.defense.gouv.fr)
*** *DGA, Vernon, France*
(e-mail: alain-p.martin@dga.defense.gouv.fr)
**** *Mines-Paris Tech, Paris, France*
(e-mail: nicolas.petit@mines-paristech.fr)

Abstract: This paper describes a technique for timestamping data from an array of sensors. This is a challenging issue that needs to be solved for the magnetometry-based navigation system presented here. This system uses finite difference schemes to evaluate spatial and time derivatives of a vector field. The accuracy of the timestamping method is critical. In the context of this particular application, we detail the various difficulties that must be circumvented. We present a software architecture, which was run on an experimental device.

Keywords: Embedded systems, discrete measurements, delay analysis, sensor fusion, control application, data fusion, inertial measurement unit, magnetic fields.

1. INTRODUCTION

In this paper, we propose a method to accurately timestamp data from a collection of sensors. This technique was developed with a view to real-time application in the field of position estimation by fusion of data from low-cost sensors. Lately, a new algorithm for indoor navigation has been proposed by Vissière et al. (2007b). This uses a set of distributed magnetometers which are combined to estimate, using Maxwell's equations (which govern the propagation of electromagnetic phenomena), the velocity of a rigid body moving in a magnetically perturbed area. This information was used in a data fusion algorithm that also makes use of inertial measurements. Typical envisioned applications range from indoor positioning systems to autonomous vehicle navigation. The collected data is used to compute estimates of spatial and time derivatives of the magnetic field. The approach can be extended to other force fields (e.g. gravitational or electric fields), requiring solely an array of corresponding sensors. In practice, estimates are obtained from finite differences of (possibly multirate) sampled data. A critical issue is the consistency of the collected data: finite difference schemes must be fed with accurate, synchronized data. More precisely, delays between the physical measurements and the computation in the real-time algorithm must be reliably known and compensated for. Otherwise, large distortions and inconsistencies will unavoidably occur and introduce disastrous discrepancies, resulting in the failure or divergence of data fusion algorithms. We propose a technique to address these issues.

This paper is organized as follows. In Section 2, we describe the distributed sensors under consideration. We discuss the limitations of classic data acquisition techniques. In Section 3, we propose an alternative method which addresses the issues discussed. The main contribution is a description of the embedded timing and data collection algorithm. In Section 4, we report experimental results, and finally, in Section 5, we give some conclusions and sketch future directions for research.

2. PROBLEM STATEMENT

2.1 Distributed magneto-inertial positioning

In most *embedded positioning devices* (see e.g. Caccamo et al. (2005)), information is repeatedly collected from the sensors at successive time steps. Data is processed in a main loop and little care is given to the time delays between the various sensors. Implicitly, those delays are assumed to be negligible. Lately, the concept of *reliable implementation of a real-time computing platform* has emerged (see Horowitz et al. (2003)). This research effort focuses on controller implementation, and proposes ways to guarantee that control algorithms can provide answers within a mandated deadline. This point has been identified as one of the main culprits for the lack of robustness in advanced control algorithms. In a complementary spirit, we focus in this paper on a rather different weakness of implementation, which stems from the measurement system itself.

We consider a data fusion algorithm in which the dates of the measurements made by the various sensors play a key role. The technique under consideration is outlined in Vissière et al. (2007a). It aims at providing navigation information for the motion of a rigid body located inside a magnetically disturbed area. A typical situation can be observed indoors: in a standard office, the Earth's magnetic field is significantly disturbed. The direction of this vector can be altered by ± 30 degrees, and its magnitude by a factor of 0.5 to 1.5 (see Figure 1). Rather than considering the magnetic measurements as too greatly disturbed to be usable, it is in fact possible to determine useful information from the disturbances themselves. We now briefly sketch how. The magnetic field satisfies an evolution equation (from Maxwell's equations)

$$\dot{M} = -\Omega \times M + R \nabla^2 h R^T V \qquad (1)$$

where M is the vector magnetic field, R is the rotation matrix of the rigid body, Ω is the vector rate of rotation of the rigid body, and $\nabla^2 h$ is a matrix determined by the spatial derivatives of the vector magnetic field. Finally, and most interestingly, V is the vector velocity of the rigid body. This quantity is the most difficult to estimate (see e.g. Faurre (1971); McClure (1960)) in inertial navigation. In turn, it is also very difficult to determine the position itself. Using an inertial measurement unit (IMU), R, M, and Ω can be estimated. Further, by considering Eq. (1), the velocity V can be estimated, provided that the spatial and time derivatives of M can be evaluated. Under the preceding assumptions, this estimation can significantly enhance the performance of any inertial navigation system, as has been demonstrated by Vissière et al. (2007a). However, a major difficulty remains. This is to reliably estimate these derivatives, and, most importantly, the coefficients of the symmetric matrix $\nabla^2 h$. In the application considered in this paper, the coordinates are obtained by Taylor first-order difference schemes. For that purpose, an array of spatially distributed magnetometers is considered, and the measurements from them are combined. There are two specific features that need special care:

- The magnetic-field values measured have to be differentiated. This amplifies the measurement errors.
- The time variations of the field are not necessarily slow compared with the sampling rate, and significant information lies in those variations.

Obtaining spatial derivatives of a field which changes quickly compels one to have very well-synchronized data. As will be shown in Section 2.3, the accuracy requirements are quite strict. To collect information from the sensors, two approaches can be considered:

- Having all the data delivered simultaneously by the sensors.
- Adding a timestamp to the data and interpolating the data values to obtain synchronized estimates of the actual sensed field, to take account of lags between the sensors.

Whereas the first approach may seem the most straightforward and efficient, it is the most difficult to put into practice. There are two main reasons for this.

First, it must be assumed that all of the sensors can be interrogated simultaneously, and that they have trusted "upon request" behavior. In other words, it must be assumed that they perform the measurement task exactly when they are asked to do so. This is not the case for most low-cost sensors. A careful investigation of the magnetometers used in the present study revealed that they continuously performed periodic measurements, and delivered the latest one whenever a data request was made. Unfortunately, these magnetometers do not supply any information about the delay between the physical measurement and the output of a value. With such sensors, data synchronization is almost impossible. The outputs of the sensors do not depend on how they are interrogated. The sensors are not synchronized, and there is not much that can be done about this. Secondly, it also must be assumed that information from all of the sensors can be read simultaneously. This task is usually performed by a microcontroller, which can be overwhelmed. This yields further synchronization errors.

A more robust way to proceed is to choose the following approach. The sensors are set to a continuous output mode in which they continuously deliver measurement information as soon as it is available (i.e. right after the physical measurement). Provided that the microcontroller reacts at the exact time when this information is received, and stores the date of reception along with the data, then there is no uncertainty in the actual timestamping of the measurements. This is the method we consider here. It is detailed in the rest of the paper.

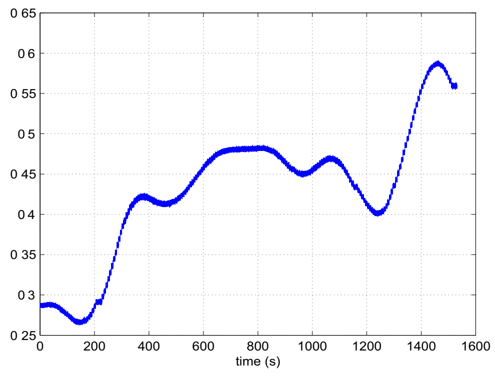


Fig. 1. Variation of the norm of the magnetic field during a 2.4 m horizontal slow move inside a standard office.

2.2 Hardware

The measurement system discussed here consisted of an inertial measurement unit (IMU), eight spatially distributed magnetometers, and a microcontroller, which was connected to a remote PC which was in charge of the data fusion algorithms. This system is pictured in Figure 2. The sensors are briefly described below; the data fusion algorithm and the overall software architecture are described in Vissière and Petit (2008). This collection of sensors delivered, through the microcontroller, a single message gathering all the measurements together, which was sent to the remote PC at a rate of 76 Hz. The message was sent through a high-speed serial port.

- **Inertial Measurement Unit**
  The IMU under consideration (Table 1) was a 3DM-GX1 from Microstrain$^{\text{TM}}$. This contained three angular-rate gyroscopes, three orthogonal single-axis magnetometers, and three single-axis accelerometers,

along with 16 bit A/D converters and a microcontroller. In our setup, the IMU was asked to deliver only temperature-compensated sensor data at a rate of 76 Hz, its fastest output rate.

Table 1. Specification of the IMU

Weight	Cost	Size (mm)	f	Communication
30 g	\$1450	39, 54, 18	76 Hz	RS232

- **Magnetometers**
  We used eight HMR2300 three-axis magnetometers from Honeywell^{TM} (Table 2). Their range was $\pm$ 2 G and they had a 70 μG resolution. They were placed on the edges and sides of a cube centered on the IMU (see Figure 2).

Table 2. Specification of the magnetometers

Weight	Cost	Size (mm)	f	Communication
28 g	\$230	75, 30, 20	154 Hz	RS232

- **MPC555 microcontroller**
  The microcontroller, which served as an interface for the sensors, was an MPC555 Power PC from Motorola^{TM} (Table 3). It ran a specific software package developed using the Phytec^{TM} development kit. It had a relatively fast 40 MHz clock.

Table 3. Specification of the microcontroller

Weight	Cost	Size (mm)	f	Communication
25 g	\$450	72, 57, 8	all	all

The microcontroller ran specific interrupt-driven software written in C. Information from each sensor was transferred using a dedicated interrupt handler routine.

The data acquisition software running on the microcontroller was event-driven, driven by the IMU messages, which were 31 bytes long. Once a message from the IMU had been received and validated by the microcontroller, information from the magnetometers was picked up in data buffers which were fed with serial messages through hardware interrupts at 154 Hz. Information was gathered together into a 116 byte message containing all of the onboard measurements and sent through a high-speed serial port. Figures 3 and 4 describe the algorithm with more accuracy. Figure 3 contains a schematic view of the interrupt management, and Figure 4 gives an overview of the main loop of the real-time embedded software.

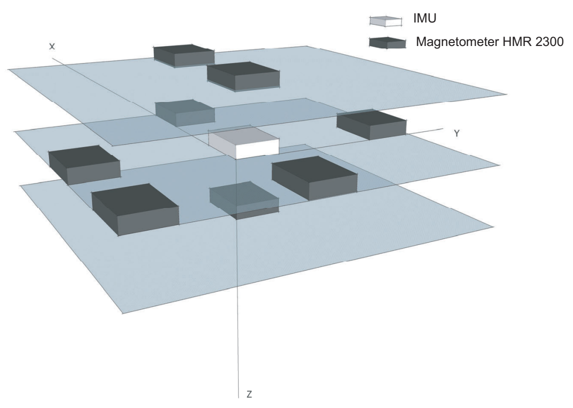


Fig. 2. Test bed: the IMU (which contains its own three-axis magnetometer) is located in the center of an array of eight magnetometers distributed on three parallel planes. The distance between the sensors pictured is about 10 cm. The power PC, the battery, and all of the embedded electronics were placed on one side of the platform and are not represented in this schematic.

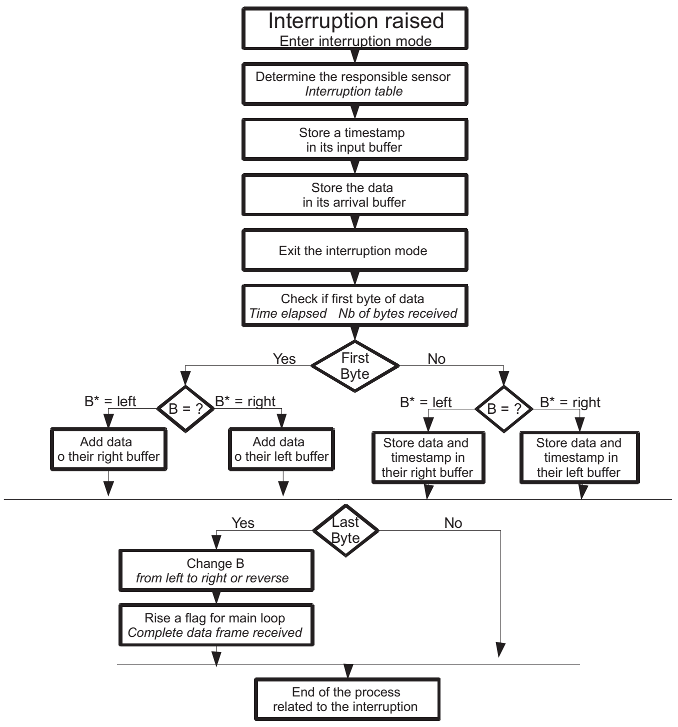
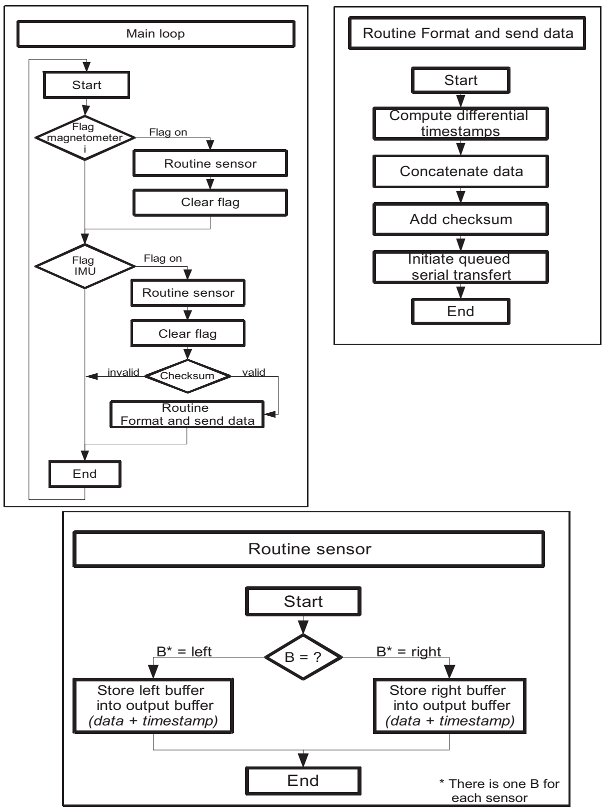

Interruption raised
Enter interruption mode
Determine the responsible sensor
Interruption table
Store a timestamp in its input buffer
Store the data in its arrival buffer
Exit the interruption mode
Check if first byte of data
Time elapsed Nb of bytes received
First Byte
Yes
No
B* = left
B = ?
B* = right
B* = left
B = ?
B* = right
Add data o their right buffer
Add data o their left buffer
Store data and timestamp in their right buffer
Store data and timestamp in their left buffer
Last Byte
Yes
No
Change B
from left to right or reverse
Rise a flag for main loop
Complete data frame received
End of the process related to the interruption

Fig. 3. Interrupt management: the interrupt mode is left as soon as possible in order to avoid delaying another interrupt. Two buffers (left and right) are used. At any given time, one is full of available data (the last complete data frame received) and ready to be read, whereas the other is being written with new data.

2.3 Limitations of simple measurement methods

In the context of multisensor data fusion, a standard approach (used in e.g. Elkaim and Foster (2006)) is to associate the latest available data from each sensor at each time step. Implicitly, it is assumed that the sensors have performed the measurements simultaneously. In most applications, timing errors are not significant, because the rate of change of the measured variables can be assumed small enough. In contrast, for our particular application, the errors are greatly amplified by the finite difference scheme. This rules out this simple method.

It is necessary to gather data from the IMU (which runs at 76 Hz) and from the eight magnetometers, running at approximately 154 Hz. Consider one magnetometer. The time elapsed between two consecutive measurement readings by the microcontroller may vary a little. This variation can be amplified by a queue of interrupts (even though the software architecture was optimized to shorten the code executed inside interrupts). This variable delay is short (an upper bound could be put on it experimentally) but it has a harmful effect. To illustrate this, the synchronization problem is pictured in Figure 6. Four possible data associations due to this uncertainty are represented. The output



Main loop
Start
Flag magnetometer i
Flag on
Routine sensor
Clear flag
Flag IMU
Flag on
Routine sensor
Clear flag
invalid
Checksum
valid
Routine Format and send data
End
Routine Format and send data
Start
Compute differential timestamps
Concatenate data
Add checksum
Initiate queued serial transfert
End
Routine sensor
Start
B* = left
B = ?
B* = right
Store left buffer into output buffer (data + timestamp)
Store right buffer into output buffer (data + timestamp)
End
* There is one B for each sensor

Fig. 4. Software main loop. Three buffers are used for each sensor: buffer_right, buffer_left, and buffer_output. The first two are used so that the last complete frame is stored in one of them while the other one is being written or ready to be written. When an IMU frame is complete, the last available data from each sensor is stored in the corresponding output buffer. This is a way to freeze all of the values that will be sent while the differential timestamps are computed and the output message is concatenated.

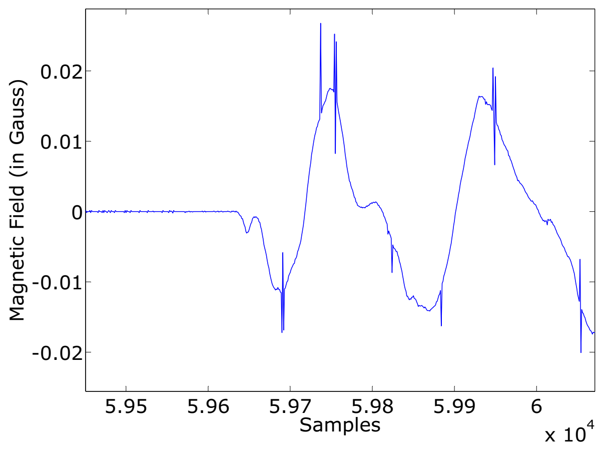


Fig. 5. Differences between measurements from the same magnetometer at two successive time steps.

rate of the IMU and the output rate of the magnetometers are nearly in an integer ratio. (The output period of the magnetometers was 6.5 ms, whereas half the period of the IMU was 6.5536 ms.) On the time line presented in Figure 6, we represent the IMU messages (IMU1, IMU2, . . .) and the messages from one of the magnetometers (M0, M1, . . .) There are some time intervals $[t_1; t_2]$ during which it is uncertain which magnetometer data corresponds to a given IMU data. Matching data are represented in red and blue, respectively. The four possible cases correspond to slight differences in the measurement delay and stress the fact that over the period $[t_1; t_2]$, the magnetometer measurements are not necessarily equally distributed over time. This phenomenon appears periodically owing to the near-integer ratio of the sensor frequencies. The unbalanced distribution of sensing intervals can be observed separately for all eight magnetometers. As a result, the collected data is hardly usable without any precise timestamping information. Fortunately, this information can be easily obtained. We now outline a solution to this problem.

3. SOLUTION METHOD

Unlike the magnetometers, which did not deliver any timestamping information, the IMU, which can be considered as a regular clock, provided accurate timing information, included in its output message. The IMU was taken as a reference clock. Next, we proceeded in two steps.

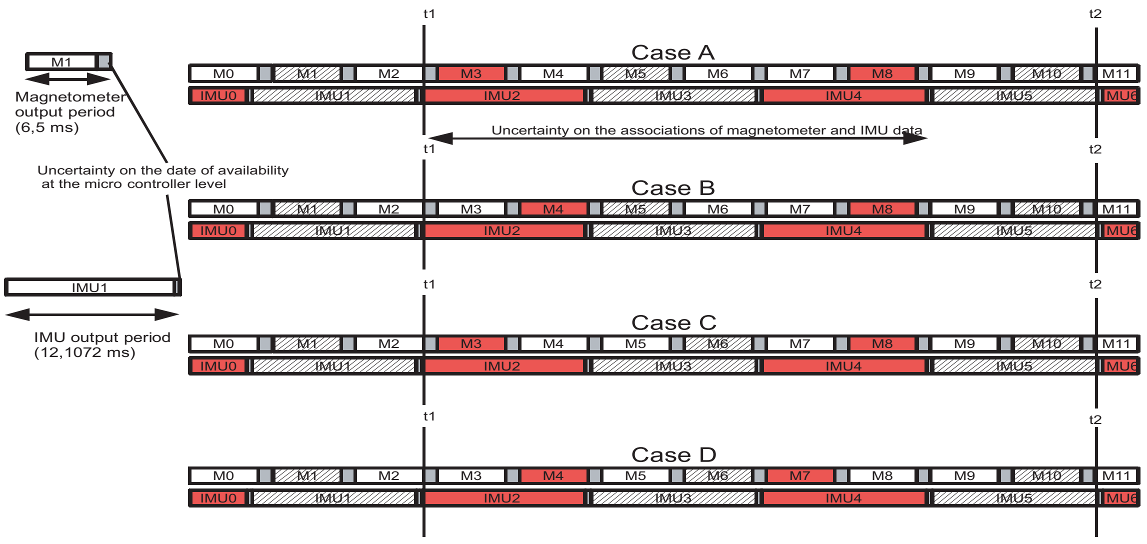
- Whenever a magnetometer measurement had been received, the microcontroller measured the time elapsed from the latest IMU measurement (see again Figures 3 and 4).
- This elapsed time was added to the IMU time and included in the gathered message.

We now give details of these two steps.

3.1 Timestamping at the microcontroller level

The reference clock was the IMU clock. The IMU tick count was included in its messages. The microcontroller clock was used to measure the elapsed time between an IMU message and the reception of measurements from the other sensors. The information from the magnetometers was obtained in the form of a message received through a serial port using dedicated interrupt handler routines. To maximize performance, the task performed inside the interrupt handler routines was limited to the following: to get the data and store the timestamp (for the first byte of data of each message only). The interrupt subroutine, which is pictured in Figure 3, was exited as soon as possible, allowing other interrupts to come into play. Further tasks were done afterwards in the main loop. These included checking the validity of the message when a checksum was available, and converting the stored acquisition timestamps into timestamps relative to the IMU clock. Finally, a single output message containing the timing information was gathered together. The main software architecture is detailed in Figure 4. The gathered message contained:

- An absolute timestamp given by the IMU reference clock.



M1
Magnetometer output period (6,5 ms)
Uncertainty on the date of availability at the micro controller level
IMU1
IMU output period (12,1072 ms)
t1
t2
Case A
Case B
Case C
Case D
Uncertainty on the associations of magnetometer and IMU data
M0 M1 M2 M3 M4 M5 M6 M7 M8 M9 M10 M11
IMU0 IMU1 IMU2 IMU3 IMU4 IMU5 IMU6

Fig. 6. Shift in acquisition time: The IMU delivers its message at 76 Hz and each magnetometer at 154 Hz, just a bit more than twice as often. The last magnetometer's data acquired when an IMU message arrives is gathered in the same output message of the microcontroller. Cases A to D are showing possible associations of magnetometer's and IMU's data.

- A relative timestamp for every magnetometer data. This represents the time difference between the reception of the magnetometer measurements by the microcontroller and the acquisition of the data from the IMU contained in the same output message.

3.2 Delay between measurement and acquisition

Physically, an additional pure delay due to the sensor itself has also to be taken into account. This delay is caused by internal computations performed at the sensor level to compensate for drift due to temperature changes, conversion of voltages into variables of interest, and transmission delays. It can be assumed that all eight magnetometers have the same delay, i.e. that no relative delay between them needs to be considered. This point is supported by an experiment where the sensors were moved quickly from one steady state to another. This results in steep variations of the measured magnetic field. Taking account of a single shared delay from the eight magnetometers and using the timestamping technique discussed above, a careful investigation of the data obtained made it clear that the time mismatch between all eight sensors was below 0.2 ms. This is visible in Figure 7.

Data sheets can provide an estimate of the absolute value of this delay, but large deviations are expected (at least for low-cost sensors). An offline identification easily yielded an estimate of this delay. In practice, measurements during large, fast rotational movements were made in a constant magnetic field. Figure 7 reports the results obtained. The magnetometers responded in a few tenths of a millisecond, and the IMU reacted slightly later. The delay between the IMU and the magnetometers could be easily determined.

Finally, note that no real, absolute time information can be obtained. The internal clock of the IMU, which was taken as a reference, is accessible after a delay which can be estimated but not completely determined. There remains a small uncertainty in the origin of the time scale.

4. RESULTS

A criterion that can be used to analyze the quality of the timestamping technique described above is its ability to picture the variation of the time delay between data received from the IMU and from each magnetometer. As explained in Section 2.3, the output rates of the magnetometers and the IMU were in a near-integer ratio: the magnetometer period (6.5 ms) was a little shorter than half the period of the IMU (6.5536 ms). Thus the time delay between the magnetometer measurements and the IMU measurements in consecutive messages decreases slowly over about a hundred samples. Then a jump appears, as depicted in Figure 6. This corresponds to the case where two successive magnetometer messages are skipped (for

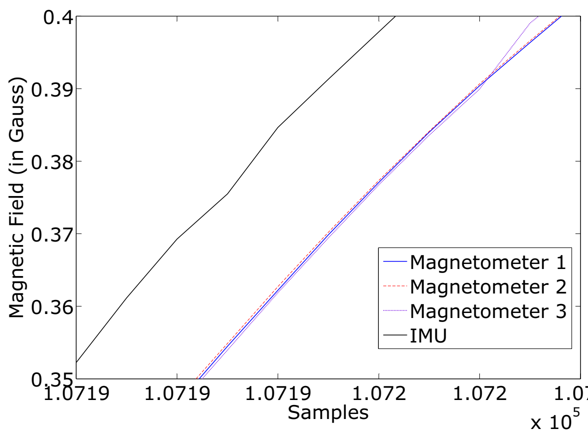


Fig. 7. A fast move from one steady state to another. This results in a steep variation of the measured magnetic field, which allows us to visualize the pure delay between sensors (once the data have been replaced at their correct positions on the time axis according to their timestamps).

instance, between M5 and M8 in case A in Figure 6). After the jump, the time delay starts decreasing once again, starting a new period. Figure 8 shows these shifts in time delays, which can be observed experimentally for three magnetometers.

Figure 9 presents results obtained with a magneto-inertial positioning algorithm which takes advantage of the technique described in this paper. The closed trajectory was that of a person walking along a loop-shaped corridor inside a standard office building and carrying the test bed described in Section 2.2.

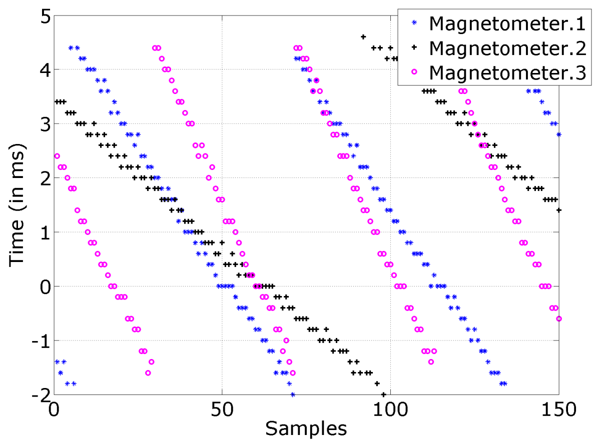


Fig. 8. Time delays between the magnetometer measurement and the IMU measurement for three magnetometers. The decreasing slopes and the quasi-period are clearly visible. Note that the slopes are not exactly the same for all magnetometers (which do not have exactly the same output rate). A very small variation of the output rate results in several samples being added to or removed from the length of the quasi-period.

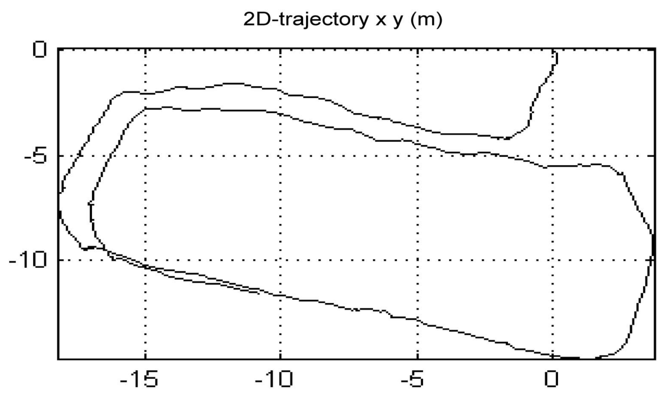


Fig. 9. Walking along a loop-shaped corridor. 2D position estimates.

5. CONCLUSIONS AND FUTURE DIRECTIONS OF WORK

The timestamping technique discussed in this paper has been successfully applied to a collection of magnetometers and an IMU. Further, it can be used in many other configurations, whenever delays between physical measurements have to be reliably known and compensated for. This is of special interest for low-cost sensors, which always have behaviors slightly different from each other.

To use low-cost sensors at their best in our magneto-inertial positioning system, another important issue has to be considered: the calibration of the set of sensors. Scale factors, and misalignments within a sensor set and between sensor sets have to be taken into account. Thanks to the proposed technique, identification of these parameters can be accurately performed.

REFERENCES

Caccamo, M., Baker, T., Burns, A., Buttazzo, G., and Sha, L. (2005). Real-time scheduling for embedded systems. In D. Hristu-Varsakelis and W.S. Levine (eds.), *Handbook of networked and embedded control systems*. Birkhäuser.

Elkaim, G. and Foster, C. (2006). Metasensor: Development of a low-cost, high quality attitude heading reference system. In *Proc. of the 19th International Technical Meeting of the Satellite Division of the ION GNSS 2006*.

Faurre, P. (1971). *Navigation inertielle et filtrage stochastique*. Méthodes mathématiques de l'informatique. Dunod.

Horowitz, B., Liebman, J., Ma, C., Koo, T., Sangiovanni-Vincentelli, A., and Sastry, S. (2003). Platform-based embedded software design and system integration for autonomous vehicles. *Proc. of the IEEE*, 91, 198–211.

McClure, C.L. (1960). *Theory of inertial guidance*. Prentice Hall.

Vissière, D., Martin, A.P., and Petit, N. (2007a). Using magnetic disturbances to improve IMU-based position estimation. In *Proc. of the 9th European Control Conf.*

Vissière, D., Martin, A.P., and Petit, N. (2007b). Using spatially distributed magnetometers to increase IMU-based velocity estimation in perturbed areas. In *Proc. of the 46th IEEE Conf. on Decision and Control*.

Vissière, D. and Petit, N. (2008). An embedded system for small-scaled autonomous vehicles. In *Proc. of the ICINCO 2008, IEEE International Conference on Informatics in Control, Automation and Robotics*.