Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2016) 24: 2010 – 2027
© TÜBİTAK
doi:10.3906/elk-1402-260

Vehicle localization systems: towards low-cost architectures

Samir SAKHI[1,*], **Abdelhafid ELOUARDI**[2], **Samir BOUAZIZ**[2], **Mahmoud BELHOCINE**[3]
[1]Digital Systems Laboratory, Military Polytechnic School, Bordj El Bahri, Algiers, Algeria
[2]University of Paris-Sud, Orsay, France
[3]Centre for Development of Advanced Technologies, Algiers, Algeria

Received: 24.02.2014 • **Accepted/Published Online:** 29.03.2014 • **Final Version:** 15.04.2016

Abstract: INS-GPS integration is a fundamental task used to enhance the accuracy of an inertial navigation system alone. However, its implementation complexity has been a challenge to most embedded systems. This paper proposes a low-cost FPGA-based INS-GPS integration system, which consists of a Kalman filter and a soft processor. Moreover, we also evaluate the navigation algorithm on a low-cost ARM processor. Processing times and localization accuracy are compared in both cases for single and double precision floating-point format. Experimental results show the advantages of the FPGA-based approach over the ARM-based approach. The proposed architecture can operate at 100 Hz and demonstrates the advantage of using FPGAs to design low-cost INS-GPS localization systems.

Key words: INS-GPS sensors integration, Kalman filter, field-programmable gate array, hardcore and softcore processor

1. Introduction

Navigation systems for land vehicles remain an interesting field for research and industrial development [1,2]. A navigation system should be able to provide reliable solutions. Its accuracy varies depending on the envisaged application. To increase the navigation system's accuracy, engineers use a high accuracy inertial measurement unit (IMU). However, the high cost and weight of a standard inertial navigation system limits its application in general areas, such as land vehicle navigation [2,3]. With the development of microelectromechanical system (MEMS) devices, MEMS-based IMUs have become low-cost systems and have proved to be a part of vehicle navigation systems [3 – 6]. Generally, this category of sensors accumulates errors during time. Therefore, they must be periodically reset using external information [1]. INS-GPS (inertial navigation system-Global Positioning System) integration could offer a reliable solution for precise localization application [1,2].

The basic navigation system loop [2] (see Figure 1) optimally combines the INS and GPS data to obtain a navigation solution with a higher updating rate and a smaller position error when compared to a navigation system that uses stand-alone sensors. An integration method is necessary to correct the localization given by the inertial navigation system (INS) by the GPS position [3].

The integration method can be achieved using a Kalman filter (KF) [4,5]. The critical time constraint of land navigation systems requires that processing of the INS data and the KF algorithm should be completed between two sensors' data-updating tasks. This imposes many constraints on the software integration, processing speed, and system architecture. Only a few papers are available on this topic [6 – 9] and they are related to the

*Correspondence: samir.sakhi@u-psud.fr

hardware development of such systems. Different methods of INS-GPS integration have been implemented to calculate the navigation solution. A loosely coupled integrated approach [3,10], implementing a GPS receiver and a low-cost strapdown INS, is used to overcome the sensors' errors.

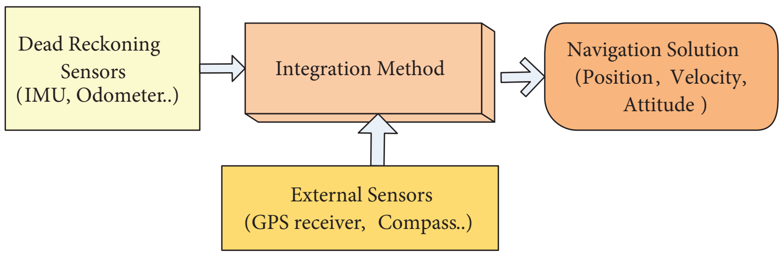

Dead Reckoning Sensors (IMU, Odometer..)
Integration Method
Navigation Solution (Position, Velocity, Attitude)
External Sensors (GPS receiver, Compass..)

Figure 1. An integrated navigation system loop.

Recent studies [6,11 − 13] have been conducted to implement the INS-GPS integrated system using platforms such as the FPGA (field-programmable gate array). The authors of [9,14] presented an INS-GPS integration on an FPGA for mobiles robotics. In [9] the implementation was achieved on an embedded processor, a Xilinx Power PC440. However, an NIOS II softcore processor was used in [14]. Other works like [12] proposed a multisensor navigation system for a 2D mobile robot using an FPGA. In this work, the authors realized a 2D navigation system using an FPGA-based embedded processor; they used a Xilinx softcore processor (Microblaze) and studied the impact of precision format on navigation system implementation. They evaluated the maximum error between single precision (SP) and double precision implementation in latitude and longitude. Islam et al. [15] proposed a design methodology for embedded navigation system design and showed the advantage of KF implementation with single precision on an embedded processor compared to those obtained from a standard desktop computer.

In this work, the INS-GPS integration algorithm is implemented on a reconfigurable architecture. To evaluate performances of the navigation system, the FPGA implementation is compared to those of a low-cost RISC processor (ARM7). The INS-GPS navigation system is decomposed into two functional blocks: the INS algorithm block and the KF block. The evaluation methodology is based on the identification of the processing tasks requiring significant computing time. It consists of several steps: we first analyze the execution time of tasks and their dependencies on the algorithm parameters. The algorithm is then partitioned in order to have functional blocks (FBs) performing defined calculations. Each block is then evaluated to determine its processing time. Functional blocks that require the most execution time are then optimized to reduce the global processing time.

This paper proposes a solution for an efficient integrated low-cost INS-GPS system for vehicle navigation. The algorithm is implemented on a reconfigurable architecture compared to a RISC processor.

The paper makes the following contributions:

- Proposition of a low-cost INS-GPS navigation system.
- A practical and efficient solution for applying the KF to real-time embedded applications.
- Flexible codesign that can be applied to other signal processing problems besides the KF.
- Comparison between FPGA implementation and ARM processor implementation in both cases of single and double precision floating-point data processing.

The paper is organized as follows: Section 2 provides the design of an integrated INS-GPS system; we rely on loosely coupled mode. We describe the navigation algorithm, namely the inertial navigation calculations, and the KF state model. Section 2 details the software implementation on a reconfigurable architecture compared to a low-cost RISC architecture. Section 3 presents the instrumented vehicle used to evaluate the navigation loop and the experimental results. Finally, conclusions and perspectives are provided.

1.1. Design of an integrated INS-GPS system

The design of a real-time navigation system is a complex task [6,15], which is brought back to adopt a data fusion approach. This system should be validated by the databases, IMU data, and GPS positions, giving the vehicle's trajectory. The navigation algorithm was decomposed into FBs to facilitate the implementation and the testing of each block [4]. We define two functional blocks: the INS block and data fusion block (Figure 2).

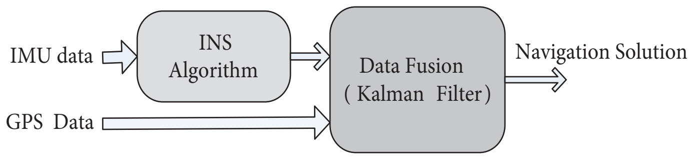


Figure 2. Architecture of the INS-GPS integrated system.

2. INS-GPS integration methods

The complexity degree of the INS-GPS integration approach should reflect the application requirements [3,16]. Integration methods can be simple or relatively complex. In our application, the navigation system integrates a public GPS receiver and a low-cost inertial sensor, which are loosely coupled (Figure 3). This mode enables a reduction of both design and implementation costs [10,17].

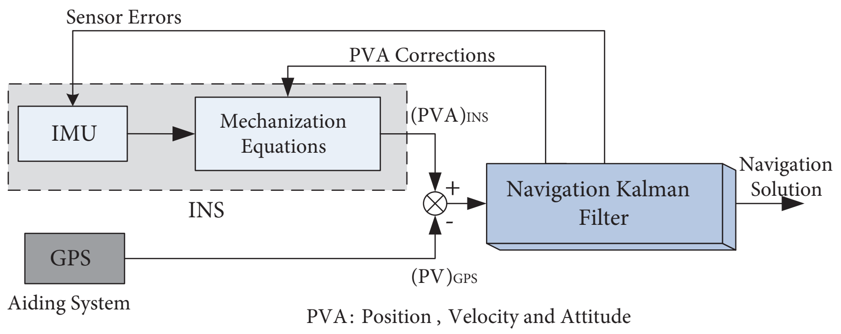


Figure 3. Loosely coupled integration approach (closed-loop).

In the loosely coupled mode an external sensor (GPS receiver) is used to correct the INS errors (Figure 3), and the GPS positions and velocities are combined with INS positions to compute the residual error, which can be exploited by the KF algorithm.

This algorithm corrects the INS state using the feedback loop, bias effects, and drifts, as well as misalignment errors. This is especially important for medium to low accuracy IMU systems [3]. According to different applications, a loosely coupled scheme can be implemented in a variety of configurations. The configurations most commonly used are open-loop and closed-loop implementations [18,19].

2.1. INS algorithm

The INS algorithm is used to achieve the INS-GPS integration. In this case we need the INS computation equations and the INS error model to define the Kalman filter model [19].

2.1.1. INS mechanization equations

In the inertial navigation system, an IMU is mounted directly within the vehicle chassis. Hence, three components of the specific force vector and three components of the angular velocity vector in the body frame denoted by f^b and w_{ib}^b respectively can be measured. The actual vehicle's position P is described by the latitude φ and longitude λ in the geographic frame and the height h of the vehicle relative to the earth ellipsoid. The equation in the navigation frame is as follows [1]:

$$\begin{bmatrix} \dot{p} \\ \dot{v}^N \\ \dot{R}_b^n \end{bmatrix} = \begin{bmatrix} v^N \\ R_b^n f^b - \left(2w_{ie}^n + w_{en}^n\right) v^N + g^n \\ R_b^n \left(\Omega_{ib}^b - \Omega_{ie}^b\right) \end{bmatrix} \tag{1}$$

where:

p: Position vector in the navigation frame, $p = [\varphi, \lambda, h]$

v^N: Velocity vector in the navigation frame, $v^N = [v_n, v_e, v_d]$

R_b^n: Transformation matrix from body frame to navigation frame

w_{ie}^n: Earth rotation rate vector expressed in the navigation frame

w_{en}^n: Orientation rate vector of the navigation frame

Ω_{ib}^b: Skew-symmetric matrix of the body rotation rate

Ω_{ie}^b: Skew-symmetric matrix of the rotation rate vector

w_{ie}^b: Rate of earth rotation expressed in the body frame

g^n: Gravity vector expressed in the navigation frame

2.1.2. INS error model

In order to navigate over large distances around the earth, navigation information is most commonly required in the local geographic frame [1,18]. The derivation of vehicle position $P = [\varphi, \lambda, h]$ is given by Eq. (2) [18]:

$$\dot{p} = \begin{bmatrix} \dot{\varphi} \\ \dot{\lambda} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} \frac{v_n}{R_M + h} \\ \frac{v_e}{\cos(\varphi)(R_N + h)} \\ -v_d \end{bmatrix} \tag{2}$$

where R_M, R_N are respectively the meridian and the normal earth radius.

Inertial navigation systems are affected by many kinds of errors; most of them are caused by initial condition errors, nonorthogonal properties of gyrometers and accelerometers, and the accumulation of instrumentation errors through the integration process [10,17]. The INS error model is developed from the mechanization approach (Eqs. (1) and (2)). Most errors are attributed to the inertial sensors (gyrometers and accelerometers). The dominant error sources are bias, scale factor error, nonorthogonality, and random noise [10,18]. In this case, the accelerometer and gyrometer measurements can be assumed as [1]:

$$\Delta f^b = f^b - \tilde{f}^b \tag{3}$$

$$\Delta w_{ib}^b = w_{ib}^b - \tilde{w}_{ib}^b \tag{4}$$

where $\tilde{f}^b$ and $\tilde{w}_{ib}^b$ are respectively the measured values of f^b and w_{ib}^b

Δf^b and Δw_{ib}^b represent specific force and relative angular rate measurement errors.

Similarly, the estimated values of the specific force error ($\Delta \hat{f}^b$) and the relative angular rate error ($\Delta \hat{w}_{ib}^b$) can be expressed by the same form [1].

Since we are trying to correct the errors in a navigation system, working in terms of error states (δx) is convenient [19].The INS model will have the following form [1]:

$$\begin{bmatrix} \delta\dot{p} \\ \delta\dot{v} \\ \dot{\rho} \end{bmatrix} = F \begin{bmatrix} \delta p \\ \delta v \\ \rho \end{bmatrix} + G \begin{bmatrix} \delta f^b \\ \delta w_{ib}^b \end{bmatrix} \tag{5}$$

with $F = \begin{bmatrix} F_{pp} & F_{pv} & F_{p\rho} \\ F_{vp} & F_{vv} & F_{v\rho} \\ F_{\rho p} & F_{\rho v} & F_{\rho} \end{bmatrix}$ and $G = \begin{bmatrix} 0 & 0 \\ -R_b^n & 0 \\ 0 & R_b^n \end{bmatrix}$

The error vector is defined by $\delta x = [\delta p, \delta v, \rho]^T$, where $\delta p = p - \hat{p} = [\delta\varphi, \delta\lambda, \delta h]^T$, $\delta v = v_e^n - \hat{v}_e^n = [\delta v_n, \delta v_e, \delta v_d]^T$, $\rho = [\in_N, \in_E, \in_D]^T$, $\delta f^b = \Delta f^b - \Delta\hat{f}^b$, and $\delta w_{ib}^b = \Delta w_{ib}^b - \Delta\hat{w}_{ib}^b$.

In [16] several forms of the matrix F that takes place in Eq. (5) are proposed. In our case a simplified error model is used based on a first-order approximation form defined by Mohinder and Andrews [5, pp. 449–452].

In this case, the subcomponents of matrix F are given by:

$$F_{pp} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \; F_{pv} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \; F_{p\rho} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (6)(7)(8)$$

$$F_{vp} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \tau_D^{-2} \end{bmatrix}, F_{vv} = \begin{bmatrix} 0 & -2w_{ie}\sin\varphi & 0 \\ 2w_{ie}\sin\varphi & 0 & 2w_{ie}\cos\varphi \\ 0 & -2w_{ie}\cos\varphi & 0 \end{bmatrix} \quad (9)(10)$$

$$F_{v\rho} = \begin{bmatrix} 0 & f_D + 2g & -f_N \\ -f_D - 2g & 0 & f_E \\ f_N & -f_E & 0 \end{bmatrix}, \; F_{\rho p} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (11)(12)$$

$$F_{\rho v} = \begin{bmatrix} 0 & R_e^{-1} & 0 \\ -R_e^{-1} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, F_{\rho\rho} = \begin{bmatrix} 0 & -w_{ie}\sin\varphi & 0 \\ w_{ie}\sin\varphi & 0 & w_{ie}\cos\varphi \\ 0 & -w_{ie}\cos\varphi & 0 \end{bmatrix} \quad (13)(14)$$

where:

φ is the latitude and $T_D \approx \sqrt{R/_{2g}} = 520\,(s)$

$R_e = \sqrt{R_M R_N} \approx 6372795.5m$ is the earth average radius

$w_{ie} \approx 7.292115 \times 10^{-5}\, {}^{rad}/_{s}$ is the earth rotate rate

f_N, f_E, f_D are north, east, and downward INS specific force, respectively

2.2. Kalman filtering

In the INS-GPS loosely coupled integration, the KF involves the combination of two variable estimates (INS and GPS data) to form the PVA (position, velocity, and attitude) solutions [17,19]. In our case, the first estimate is provided directly by the inertial navigation system, which constitutes the process model. The second estimate, i.e. the measurement, is provided by GPS.

For a navigation system, the state of the system (vehicle) is naturally described by position, velocity, and attitude (PVA solutions) [19]. Given a plant (dynamic system) described by linear system equations in continuous state space [5]:

$$\dot{x}(t) = F(t)x(t) + G(t)u(t) \tag{15}$$

where $F(t)$ is the dynamic matrix (obtained by partial derivatives), $x(t)$ is the state vector, $G(t)$ is the design matrix, and $u(t)$ is the forcing function. The elements of $u(t) = \left[\delta f^b, \delta w_{ib}^b\right]^T$ are white noise whose covariance matrix is given by $Q = diag\left(\sigma_{ax}^2, \sigma_{ay}^2, \sigma_{az}^2, \sigma_{wx}^2, \sigma_{wy}^2, \sigma_{wz}^2\right)$.

The measurement model is given by:

$$z(t) = H(t)x(t) + v(t) \tag{16}$$

where $z(t)$ is the measurement at time t, H is the observation matrix, and $v(t)$ is the white noise with $v(t) \sim N(0, R)$.

However, for the implementation of INS, because the sampling time interval $\Delta t = t_k - t_{k-1}$ is very small (update rate of INS = 100 Hz), the plant (vehicle's movement: PVA variation vector) and the measurement model take the form summarized in Table 1 [17,18].

Table 2 summarizes the discrete-time KF equations [1,5].

2.2.1. Kalman filter implementation

One of the implicit assumptions in the KF computation is that the arithmetic algorithm uses infinite precision numbers [1]. However, the hardware implementation needs finite precision data (especially floating-point). Moreover, the KF performances can be affected by computer round-off and the unchecked error propagation in inverse matrix calculation [20]. After analysis of KF equations, it seems that computational complexity is summarized in the gain calculation (see Table 2), since it implements matrix inversion. The error covariance matrix P is by its definition positive semidefinite. Computed values of P can lose the positive semidefinite property due to finite precision of computer and round-off errors. When this occurs, the KF gains for the corresponding states may temporarily diverge [1,20]. Explicit inversion of a full matrix needs intensive arithmetic operations. A solution to reduce this complexity is to convert this problem into an easy decomposition problem, which will result in analytic simplicity and computational convenience [21]. The numerical instability of the inverse matrix calculation can be solved by factorization methods. Resulting KF implementations are called square-root filtering [5].

Table 1. Plant and measurement models.

Plant model
$x_k = \Phi_{k-1}x_{k-1} + w_{k-1}$, $w_k \sim N\left(0, Q_k\right)$
where $x_k = \left[\Delta x, \Delta y, \Delta z, \Delta V_N, \Delta V_E, \Delta V_D, \in_N, \in_E, \in_D\right]^T$
$\Phi_k = \exp\left(F\Delta t\right) \approx I + F\Delta t :$ *State transition matrix*
F: Matrix defined in Eq. (5)
$\Delta t = t_k - t_{k-1}$: difference between the time interval
Measurement model
$z_k = H_k x_k + v_k$
$Z_k = \begin{bmatrix} \left(R_e + h\right)\left(\varphi_{INS} - \varphi_{GPS}\right) \\ \left(R_e + h\right)\cos\varphi\left(\lambda_{INS} - \lambda_{GPS}\right) \\ h_{INS} - h_{GPS} \end{bmatrix} + v_k$
$H_k = \left\langle \begin{array}{ccc\|cc} \left(R_e + h\right) & 0 & 0 & & \\ 0 & \left(R_e + h\right)\cos\varphi & 0 & 0_{3\times3} & 0_{3\times3} \\ 0 & 0 & 1 & & \end{array} \right\rangle$, $and\, R = diag\left(\sigma_\varphi^2, \sigma_\lambda^2, \sigma_h^2\right)$
Where z_k: Measurment vector at time instant k and $v_k \sim N\left(0, R_k\right)$
φ, λ, h: latitude, longitude and height.
$\sigma_\varphi, \sigma_\lambda, \sigma_h$:Standard deviations of the GPSmeasurements.
R_e: Earth average radius.

In the literature, there are several matrix inversion methods based on factors decomposition. The most used methods are Cholesky decomposition, LU, and QR decomposition. In the first two methods (Cholesky and LU decomposition), the matrix to be inverted must be positive definite. On the other hand, no condition is imposed for the QR method [21].

2.2.2. QR decomposition method

The QR decomposition method allows the factorization of the matrix to be inverted into a product of two matrices: a triangular and an orthogonal matrix. Primarily, the triangular matrix inversion requires fewer calculations compared to a full matrix inversion. Secondly, the orthogonal matrix inversion is evaluated by the transpose operation [21]. Given a matrix $A \in \Re^{n\times n}$, QR factorization exists as:

$$A = Q \times R \tag{17}$$

where $Q \in \Re^{n\times n}$ is an orthogonal matrix and $R \in \Re^{n\times n}$ is an upper triangular matrix.

Table 2. Discrete-time KF equations.

Initialization	$\hat{x}_0^- = E\left[x_0\right], P_0^- = var\left(x_0^-\right)$
Gain calculation	$K_k = P_k^- H_k^T\left(R_k + H_k P_k^- H_k^T\right)^{-1}$
Measurement update	$\hat{x}_0^+ = \hat{x}_0^- + K_k\left(\tilde{y}_k - \hat{y}_k\right)$
Covariance matrix update	$P_k^+ = \left[I - K_k H_k\right] P_k^-$
Time propagation	$\hat{x}_{k+1}^- = \Phi_k \hat{x}_k^- + G_k u_k$ $P_{k+1}^- = \Phi_k P_k^+ \Phi_k^T + Q_{dk}$ $\left(Q_{dk} = G_k Q_k G_k^T \Delta T\right)$
where x^-, x^+: a Prior and a Posterior state vector, P^-, P^+: a Prior and a Posterior error covariance matrix. $G = \begin{bmatrix} 0 & 0 \\ -R_b^n & 0 \\ 0 & R_b^n \end{bmatrix}$ and $Q = diag\left(\sigma_{ax}^2, \sigma_{ay}^2, \sigma_{az}^2, \sigma_{wx}^2, \sigma_{wy}^2, \sigma_{wz}^2\right)$ with σ_a and σ_w are respectively,the standard deviations of the accelerometers and gyroscopes.	

The inversion of the matrix A is described as follows:

$$A^{-1} = R^{-1} \times Q^T \tag{18}$$

The matrix inversion is calculated by the QR decomposition in three parts: QR decomposition, matrix inversion for R matrix, and matrix multiplication (Table 3). QR decomposition can be divided into three different methods: Gram–Schmidt orthonormalization (classical or modified), Givens rotations (GR), and Householder reflections. The MGS (modified Gram–Schmidt) method is numerically more accurate and more stable than the other QR decomposition methods [21].

Table 3. QR decomposition algorithm (QRD-MGS).

	Resulting matrices
1 for $i = 1 : n$ **2** $X_i = A_i$ **3** $for\ i = 1 : n$ **4** $R_{ii} = \lVert X_i \rVert$ **5** $Q_i = X_i / R_{ii}$ **6** for $j = i + 1 : n$ **7** $R_{ij} = \langle Q_i, X_j \rangle$ **8** $X_j = X_j - R_{ij} Q_i$	$Q = \begin{bmatrix} Q_{11} & ... & Q_{1n} \\ \vdots & ... & \vdots \\ Q_{n1} & ... & Q_{nn} \end{bmatrix}$ $R = \begin{bmatrix} R_{11} & R_{12} & \dots & R_{1n} \\ 0 & R_{22} & \cdots & R_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & R_{nn} \end{bmatrix}$

3. System implementation on low-cost architectures

The research community has developed numerous navigation systems in the last years. Several studies presented optimizations with different approaches. However, they did not explore a global optimization from the algorithmic development level to the system hardware level. In some applications, such as outdoor navigation, we would benefit from low-cost sensor technology and localization implementations on compact architectures.

In this study, we present a solution for the localization algorithm implementation taking into account these constraints. The localization system is based on a codesign approach of the INS-GPS integration [4,15]. We have conducted experiments with an instrumented vehicle. We obtained the results to demonstrate that our approach, based on low-cost sensors, a reconfigurable architecture, and an optimized algorithm, is suitable to design embedded systems for real-time navigation applications.

The reconfigurable hardware provides a system design with lower risk and allows maximum flexibility [4]. The development cycle must be guided by a design methodology that consists of adopting a modular and distributed architecture. A key aspect to achieve a high-performance circuit implementation is to ensure an efficient mapping of the algorithm onto the FPGA circuit [15]. This may involve developing a hardware architecture in which we separate the communication parts and computing parts. The latter will be decomposed into FBs, where the independent operations are performed in parallel.

3.1. INS implementation

We have chosen a reconfigurable architecture based on an FPGA implementing a soft-processor (NIOS II) [22], and we compared this to a RISC architecture based on an ARM processor (ARM7). The INS algorithm is implemented on both architectures.

Table 4 shows the acquisition time of inertial data and PVA calculations. The two architectures are running at the same frequency (100 MHz).

Table 4. Execution times of data acquisition and PVA calculations.

	Execution times (ms)	
	NIOS II processor	ARM7 processor
Data acquisition	3.793	3.756
PVA calculations	0.067	0.149
Total	3.860	3.905

According to the results of Table 4, the IMU data acquisition time is considerable (around 3.8 ms) due to the serial transmission of the IMU messages (in our case: 43 bytes at 115,200 bauds). On the other hand, the time of PVA calculations in both cases (NIOS II and ARM7) does not exceed 0.15 ms.

The NIOS II could process the PVA calculations at 14.9 K/s, two times faster than the ARM7 processor. The two platforms were running at 100 MHz (Figure 4).

The resource usage of the INS data processing in the FPGA represents 58% of I/O, 43% of logic elements, and 16% of multipliers. We use a Cyclone II 35K Logic Elements ALTERA circuit. These resources include the NIOS II processor implementation [22].

3.2. Kalman filter implementation

We have implemented the KF on the ARM7-based architecture and FPGA architecture using a NIOS II processor. We have evaluated the processing time of each functional block of the KF. Table 5 presents results of this evaluation at 100 MHz.

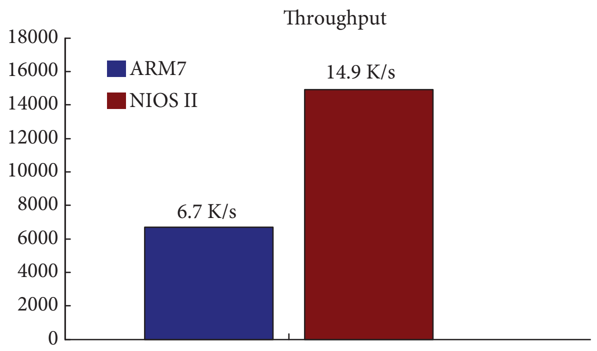


Figure 4. PVA data-flow on both NIOS II and ARM 7 processors.

According to the results of Tables 5 and 6, the hardware implementation using a soft-processor realizes a better processing time compared to the ARM7 processor. This performance is achieved by using the on-chip FPU (floating-point unit) [22].

Table 5. Processing times of functional blocks of KF in SP implementation.

	Processing times (ms)	
	NIOS II processor	ARM7 processor
Gain calculation	0.88	1.838
Measurement update	0.023	0.058
Covariance matrix update	0.495	1.163
Time propagation	1.433	3.042
Total	2.832	6.104

Table 6. Processing times of functional blocks of KF in DP implementation.

	Processing times (ms)	
	NIOS II processor	ARM7 processor
Gain calculation	0.942	2.887
Measurement update	0.025	0.089
Covariance matrix update	0.552	2.069
Time propagation	1.885	4.158
Total	3.404	9.203

To compare these results with other works, we calculate and present results for single precision (SP) and double precision (DP) floating-point implementation as well for both NIOS II and ARM7 (Figure 5). The NIOS II processor can process, in single precision, 0.16 K features/s of the KF cycle. This performance is better than the results obtained in [9] where the KF epoch is 0.13 K feature/s. The filter is implemented on an ARM920T operating at 200 MHz for mobile robot applications.

3.3. Navigation system implementation

The final step in the navigation system design is the integration of building blocks (soft COTS: component on the shell): the INS implementation block and the data fusion block based on a KF. The integration must take into account the synchronization and data flow problems.

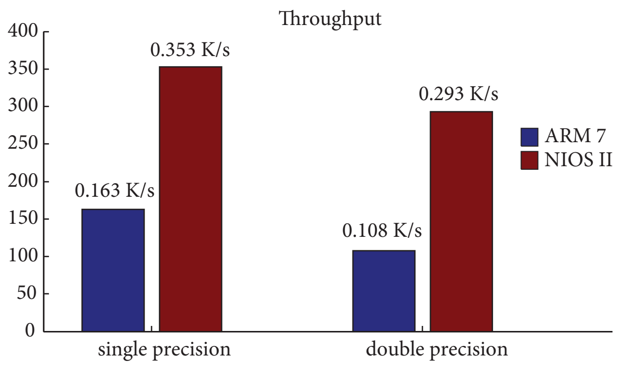


Figure 5. KF data-flow comparison between NIOS II and ARM 7 processor in SP and DP floating-point implementation. Initialization.

3.3.1. Proposed architecture

The software flow of the INS-GPS integration algorithm has several subtasks. The main program waits in an infinite loop and is based on interrupts. Figure 6 shows the various stages of computation to obtain the navigation solution (PVA vector).

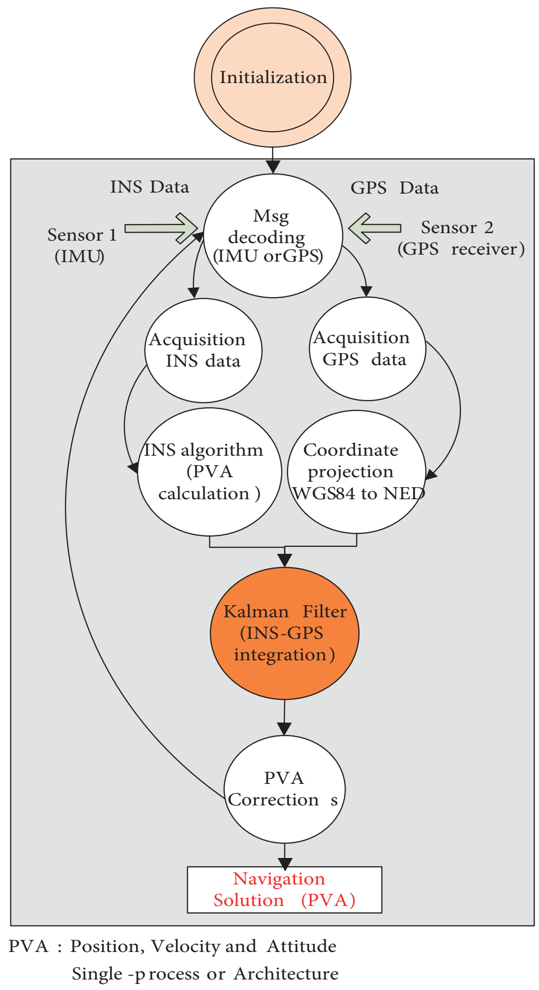

Initialization
INS Data
GPS Data
Sensor 1 (IMU)
Msg decoding (IMU orGPS)
Sensor 2 (GPS receiver)
Acquisition INS data
Acquisition GPS data
INS algorithm (PVA calculation)
Coordinate projection WGS84 to NED
Kalman Filter (INS-GPS integration)
PVA Correction s
Navigation Solution (PVA)
PVA : Position, Velocity and Attitude
Single -p rocess or Architecture

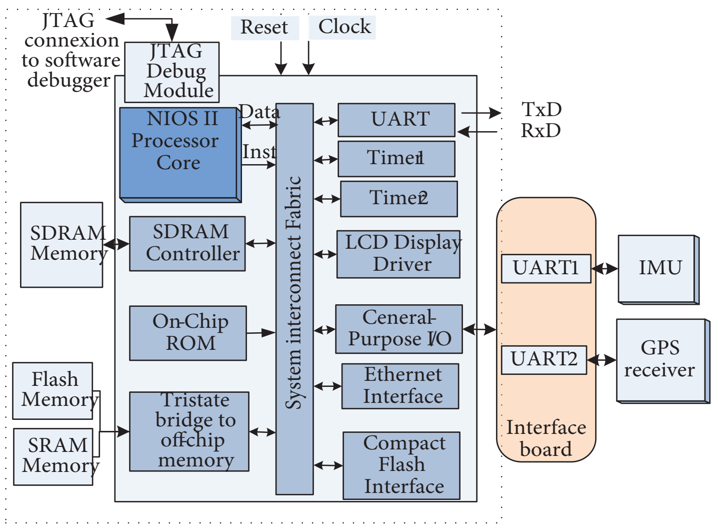
Figure 6. Block diagram of the navigation algorithm.

The INS-GPS integration algorithm is implemented in both the FPGA-based architecture (using a NIOS II soft-processor) and the ARM7-based architecture. The main specifications of the INS-GPS integration system implemented on the FPGA platform are listed in Table 7.

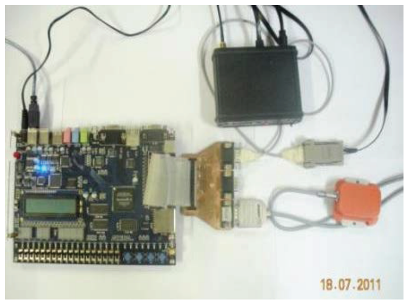
Table 7. Specification of INS-GPS navigation system.

INS	MTi of Xsens, other IMU devices are available
GPS	UBlox 4, other receivers are available
Power	12 VDC
Interface	Two 9-pin RS232 ports
Hardware	Altera DE2 Board
Software	Processor NIOS II, customized C modules including device initialization and control, message decoding, INS calculation, KF calculation, etc.

The hardware architecture of the real-time embedded navigation system is shown in Figure 7a. A test-bench platform implementing an FPGA is shown in Figure 7b.



(a). Block diagram of the navigation system



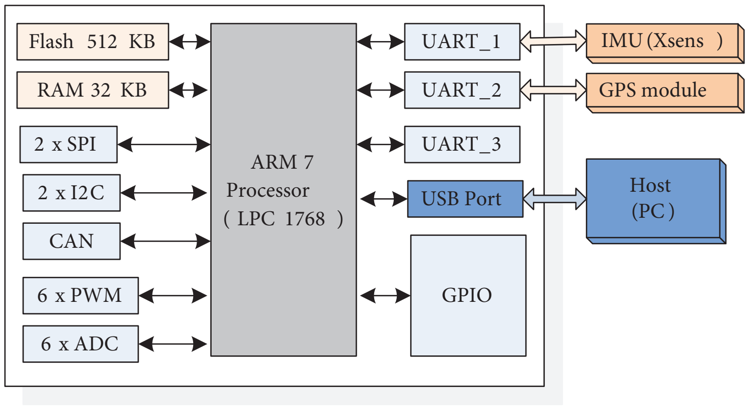
(b). Hardware evaluation system

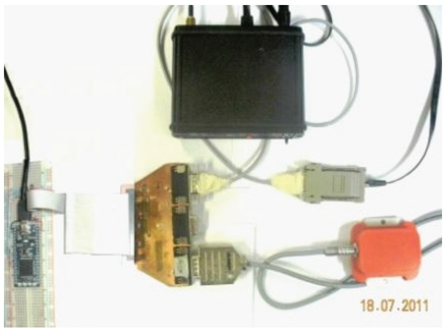
Figure 7. INS-GPS evaluation system based on NIOS II processor: a) block diagram of the navigation system,b) hardware evaluation system.

The ARM7-implementation uses the mbed board based on the NXP-LPC1768 module, with a 32-bit ARM Cortex-M3 RISC processor (Figure 8) [23].

4. Experimental results

The sensors are embedded on an experimental vehicle called “PICAR” (Figure 9) used to validate data fusion methods for autonomous vehicle applications [24]. The IMU is mounted at the center of the car and the GPS receiver is placed at the top of the car. During the experiment, we set the measurement frequency at 100 Hz for the IMU and 1 Hz for the GPS.





(a). Block diagram of the naviga

(b). Hardware evaluation system

Figure 8. INS-GPS evaluation system based on ARM7 processor: a) block diagram of the navigation system, b) hardware evaluation system.

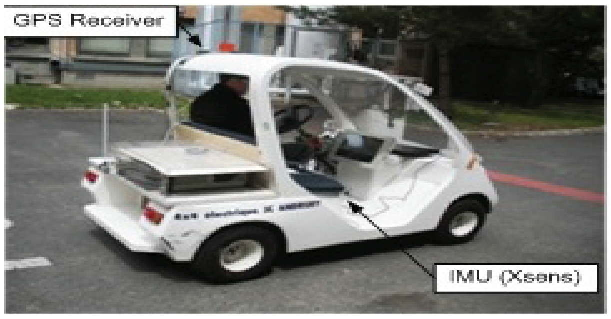


Figure 9. PICAR vehicle.

4.1. System evaluation

Two types of data collection and processing tests were performed: static and kinematic tests. These tests were carried out in order to evaluate the computation performances. The first one is the static INS test; the geographic frame [10] has been used to plot the vehicle's trajectory.

4.1.1. Static test

All types of IMU exhibit biases, scale factor, and cross-coupling errors and random noise to a certain extent [10]. The main error sources contributing to the accelerometer and gyrometer outputs are described in the following equations [10]:

$$\tilde{f}_{ib}^{i} = b_a + (I_3 + M_a)\, f_{ib}^{b} + w_a \tag{19}$$

$$\tilde{w}_{ib}^{i} = b_g + (I_3 + M_g)\, w_{ib}^{b} + w_g \tag{20}$$

where $\tilde{f}_{ib}^{b}$ and $\tilde{w}_{ib}^{b}$ are the IMU output-specific force and angular rate vectors, f_{ib}^{b} and w_{ib}^{b} are the true counterparts, and I_3 is the identity matrix. M_a and M_g are the scale factor and cross-coupling error matrix, b_a and b_g are bias, and w_a and w_g are random noise.

In our application, the used IMU (MTi: motion trackers' inertial) is already factory calibrated and it was delivered with an MT test and calibration certificate. Therefore, it remains to estimate the bias and noise variance for the accelerometer and the gyrometer by the static test.

The INS static test is carried out at a fixed GPS position of $48°41'\ 53.78$ N, $2°10'1.32$E and altitude of 142 m. Inertial measurements records of 15 min were achieved with a fixed vehicle position (reference position). The estimated biases of accelerometer and gyrometer are given in Table 8.

After the elimination of static bias, by subtracting IMU measurements, we estimated the noises (w_a, w_g) of accelerometers and gyrometers. Table 9 shows the variances of noise measurements.

Table 8. Accelerometer and gyrometer bias estimates.

	Axis	x	y	z
Bias (m/s^2, rad/s)	Accelerometer	0.0254	0.0403	9.8056
	Gyrometer	0.0017	0.0032	0.0044

Table 9. Variances of IMU random noises.

	Axis	x	y	z
Variance ((m/s^2)^2, (rad/s)^2)	Accelerometer	3.999×10^{-4}	0.0010	6.717×10^{-4}
	Gyrometer	9.177×10^{-5}	5.465×10^{-5}	3.781×10^{-5}

4.1.2. Prefiltering IMU data

Data produced by the IMU are extremely noisy. Two filters were thus designed and tested: a moving average filter [25] and a scalar filter (also called scalar gain interpretation) [26]. The choice of the prefilter has an apparent effect on the vehicle localization.

According to Figures 10a and 10b, the best performances are achieved by using the scalar filter. This filter reduces both sensor noises and vehicle vibration effects. In Figure 10, we note that the experimentation takes 43 s on a circular trajectory; however, the INS positioning (in blue in Figure 10b) gives satisfactory results until 24 s. The position and the orientation of the INS diverge after 24 s.

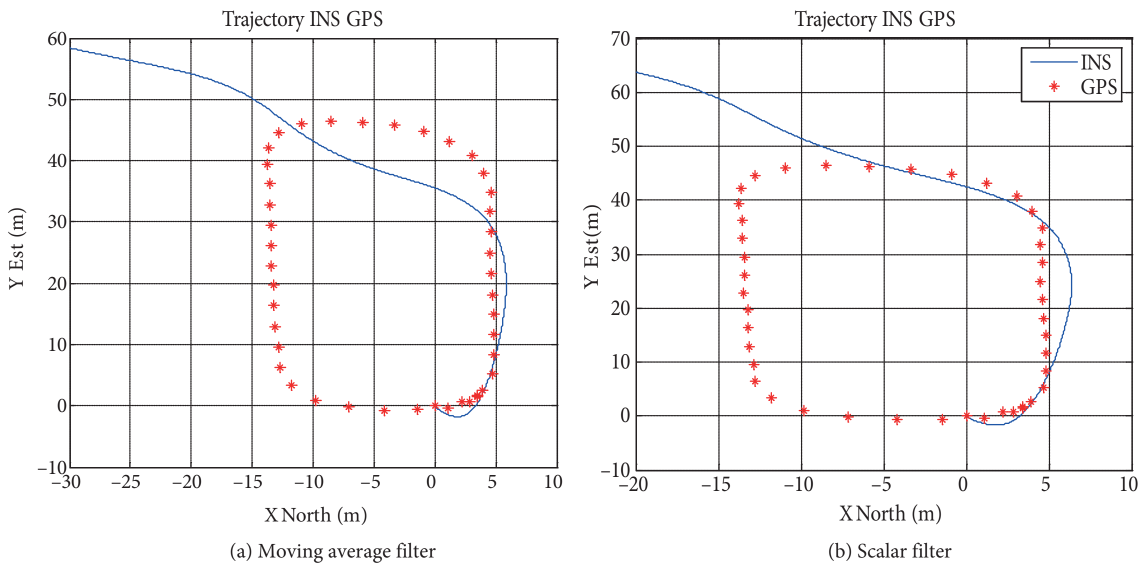


Figure 10. Comparison of two INS prefilters: a) moving average filter, b) scalar filter.

4.1.3. Kinematic test

The second test is carried out in an urban area to check the efficiency of the INS-GPS integration algorithm in real navigation conditions (Figure 11).

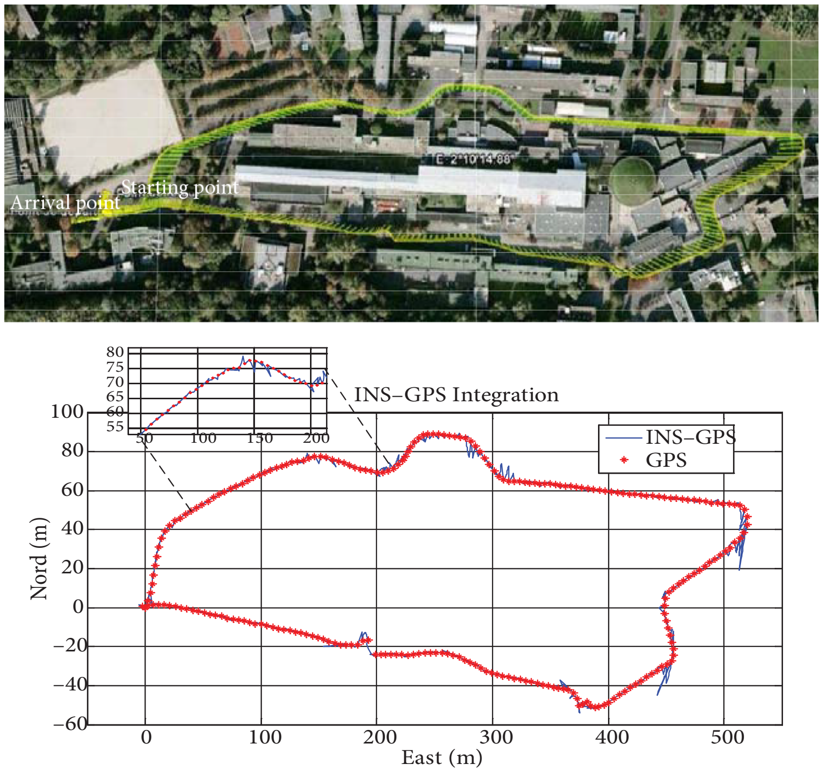


Figure 11. Trajectory of the experimental vehicle: a) real trajectory, b) estimated trajectory using INS-GPS integration.

The KF adjustment is made by the two covariance matrices, Q and R, with:

$$Q = diag\left(\sigma_{ax}^2, \sigma_{ay}^2, \sigma_{az}^2, \sigma_{wx}^2, \sigma_{wy}^2, \sigma_{wz}^2\right) \tag{21}$$

σ_a and σ_w are respectively the standard deviations of the accelerometers and gyrometers calculated in the static test.

R depends on various factors including receiver technology, antenna technology, and the user environment [17] . In our experiment, the matrix R is given by [16] :

$$R = \frac{PDOP^2 \times UERE^2}{3} I \tag{22}$$

where $UERE$ is the user equivalent range error, representing the GPS mean error (in our case $UERE = 3$) [16]; I is the identity matrix with the appropriate dimension; and $PDOP$ is the position dilution of precision, from the GPS message.

The navigation system is tested kinematically in an urban area (Paris-Sud University) with an experiment lasting 7 min. The KF adjustment is made by the two covariance matrices Q and R (Eqs. (21) and (22)).

Figure 11a presents the trajectory of the experimental vehicle in the kinematic test, and Figure 11b presents the estimated trajectory of the vehicle.

Another test was carried out to evaluate the impact of data floating-point format on the trajectory precision. We achieved this evaluation on the NIOS II soft-processor (Figure 12). We reconstructed the vehicle's trajectories based on single precision (SP) and double precision (DP) floating-point format in the processing phase of the navigation algorithm.

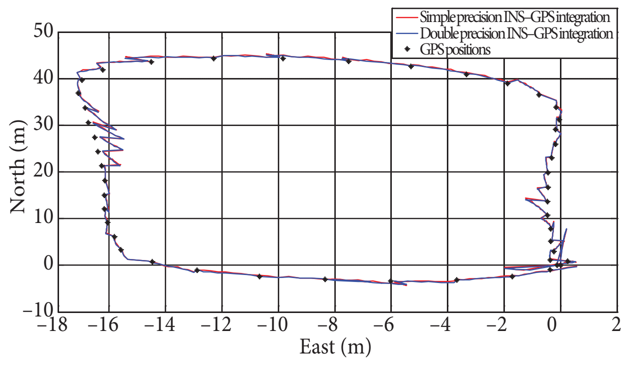


Figure 12. SP and DP calculated trajectories by NIOS II processor.

As can be seen in Figure 12, DP implementation gives a more accurate position than SP implementation.

Table 10 evaluates the error of the trajectory reconstruction relatively to the SP or DP floating-point format in the algorithm processing. Table 10 shows that double precision INS-GPS algorithm implementation increases the localization position error by 0.197 m using the NIOS II soft-processor and by 0.25 m using the ARM7 [27] processor architecture.

Table 10. Error position between SP and DP implementations.

	NIOS II processor	ARM7 processor
Min error (m)	0.003	0.006
Max error (m)	0.398	0.500
Root mean-square (m)	0.197	0.251

The proposed navigation system is able to perform the calculations at a frequency higher than the faster sensor (IMU at 100 Hz). We represent the cycle computation in a period of 10 ms (equivalent at 100%). The executed tasks (in single and double precision) during this cycle are shown in Figure 13.

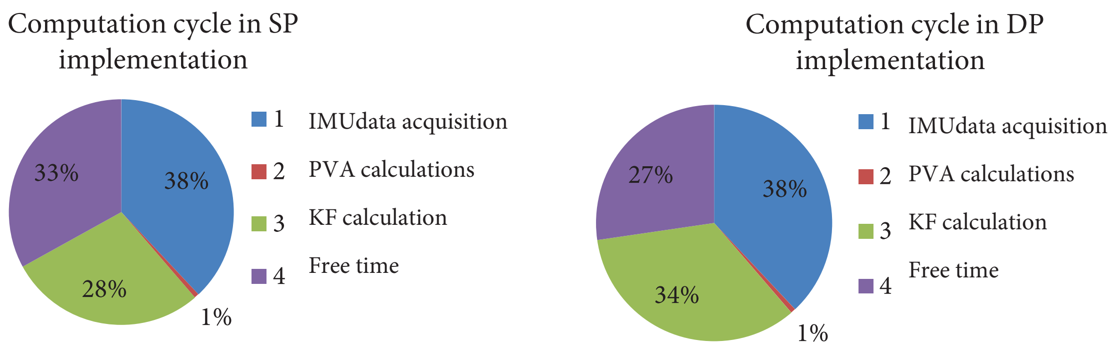


Figure 13. Executed tacks in NIOS II processor (at 100 MHz).

In both cases (SP and DP implementation), the NIOS II processor has free time around 30% of the computation cycle (10 ms).

To compare our navigation system with other works, the authors of [15] proposed a GPS/INS navigation system on an embedded platform for automotive applications. The software was executed by the Microblaze softcore processor of Xilinx. The sensor's data come from the GPS unit and a tactical-grade TG6000 IMU, which provides measurements at an update rate of 75 Hz. In this work, the Microblaze processor, at 100 MHz, can support a data rate of 238 Hz; however, in our proposed system, at the same frequency (100 MHz), the NIOS II processor can support a data rate of 345 Hz in SP implementation and 288 Hz in DP implementation.

5. Conclusions

This paper presented the implementation of a navigation algorithm by integrating data from two sensors, INS and GPS, taking into account the real-time constraints. The navigation system developed is able to acquire IMU and GPS receiver data flows and process the INS-GPS integration algorithm based on a KF. We studied the relevant parameters influencing the total performances. We used an FPGA and ARM platform to implement the same navigation system and compared the results.

We evaluated these systems on an instrumented vehicle. That gave us a database of relevant trajectories. We compared the performances and the computation precision between various hardware architectures according to the floating-point format of calculation (SP and DP). We demonstrated that the Altera NIOS II softcore processor implemented on the FPGA has much better precision performance when using the on-chip FPU IP than the ARM7's RISC Processor in this navigation system application.

References

[1] Farrell J. Aided Navigation GPS with High Rate Sensors. New York, NY, USA: McGraw-Hill, 2008.

[2] Rezaei S, Sengupta R. Kalman filter-based integration of DGPS and vehicle sensors for localization. IEEE T Contr Syst T 2007; 15: 1080-1088.

[3] Skog I. A low-cost aided inertial navigation system for vehicle applications. MSc, Royal Institute of Technology, Stockholm, Sweden, 2005.

[4] Quinchia AG, Ferrer C. A low-cost GPS&INS integrated system based on a FPGA platform. In: International Conference on Localization and GNSS; 29–30 June 2011; Tampere, Finland. pp. 152-157.

[5] Mohinder GS, Andrews AP. Kalman Filtering: Theory and Practice Using MATLAB. 3rd ed. New York, NY, USA: Wiley & Sons, 2008.

[6] Li Y, Mumford P, Rizos C. Performance of a low-cost field re-configurable real-time GPS/INS integrated system in urban navigation. In: IEEE/ION Position, Location and Navigation Symposium; 5–8 May 2008; Monterey, CA, USA. pp. 878-885.

[7] Agarwal V, Arya H, Bhaktavatsala S. Design and development of a real-time DSP and FPGA-based integrated GPS-INS system for compact and low power applications. IEEE T Aero Elec Syst 2009; 45: 443-454.

[8] Ghorbel, A, BenAmor N, Jallouli M, Amouri L. A HW/SW implementation on FPGA of a robot localization algorithm. In: 9th International Multi-Conference on Systems, Signals and Devices; 20–23 March 2012; Chemnitz, Germany. pp. 1-7.

[9] Bonato V, Marques E. A floating-point extended Kalman filter implementation for autonomous mobile robots. In: International Conference on Field Programmable Logic and Applications; 27–29 August 2007; Amsterdam, the Netherlands. pp. 576-579.

[10] Groves PD. Principles of GNSS, Inertial and Multisensor Integrated Navigation Systems. Boston, MA, USA: Artech House, 2008.

[11] Jew M, El-Osery A. Implementation of an FPGA-based aided IMU on a low-cost autonomous outdoor robot. In: IEEE/ION Position Location and Navigation Symposium; 4–6 May 2010; Indian Wells, CA, USA. pp. 1043-1051.

[12] Abdelfatah WF, Georgy J, Iqbal U, Noureldin A. 2D mobile multi-sensor navigation system realization using FPGA based embedded processors. In: 24th IEEE Canadian Conference on Electrical and Computer Engineering; 8–11 May 2011; Niagara Falls, Canada. pp. 1218-1221.

[13] Abdelfatah WF, Georgy J, Iqbal U, Noureldin A. FPGA-based real-time embedded system for RISS/GPS integrated navigation. Sensors 2012; 12: 115-147.

[14] Bonato V, Peron R, Wolf DF, DeHolanda JAM, Marques E, Cardoso JMP. An FPGA implementation for a Kalman filter with application to mobile robotics. In: International Symposium on Industrial Embedded Systems; 4–6 July 2007; Lisbon, Portugal. pp. 148-155.

[15] Islam A, Langlois JMP, Noureldin A. A design methodology for the implementation of embedded vehicle navigation systems. In: IEEE International Conference on Electro/Info Technology; 7–9 June 2009; Windsor, Canada. pp. 297-300.

[16] Abuhadrous I. Système embarqué temps réel de localisation et de modélisation 3D par fusion multi capteur. PhD, École des mines de Paris, 2005 (in French).

[17] Tan TD, Ha LM, Long NT, Duc ND, Thuny NP. Land-vehicle MEMS INS/GPS positioning during GPS signal blockage periods. VNU Journal of Science Mathematics – Physics 2007; 23: 243-251.

[18] Titteron DH, Weston JL. Strapdown Inertial Navigation Technology. 2nd ed. New York, NY, USA: IEE, 2004.

[19] Zhao, Y. Key technologies in low-cost integrated vehicle navigation systems. PhD, Royal Institute of Technology, Stockholm, Sweden, 2013.

[20] Özbek L, Köksal BE, Murat EFE. Stochastic stability of the discrete-time constrained extended Kalman filter. Turk J Electr Eng Co 2010;18: 211-224.

[21] Irturk A, Benson B. GUSTO: An automatic generation and optimization tool for matrix inversion architectures. ACM T Embed Comput Syst 2010; 9: 32.

[22] Altera Corporation. NIOS II Processor Reference Handbook. San Jose, CA, USA: Altera, 2011.

[23] Toulson R, Wilmshurst T. Fast and Effective Embedded Systems Design Applying the ARM mbed. 1st ed. New York, NY, USA: Elsevier, 2012.

[24] Bouaziz S, Reynaud R, Larnaudie B, Elouardi A. Experimental platform car for automatic control applications. In: IEEE Information and Communication Technologies International Symposium, 3–5 April 2007; Fes, Morocco.

[25] Karthik KP, Rangababu P, Sabat SL, Nayak J. System on chip implementation of adaptive moving average based multiple-model Kalman filter for denoising fiber optic gyroscope signal. In: International Symposium on Electronic System Design; 19–21 December 2011; Kerala, India. pp. 170-175.

[26] Junkyu L, Kwangjin K, Chan GP. A study of INS/CDGPS integration with a scalar adaptive filter. In: International Conference on Control, Automation and Systems; 17–20 October 2007; Seoul, South Korea. pp. 2133-2137.

[27] Zhang W, Jiang M. Design of vehicle positioning system based on ARM. In: International Conference on Business Management and Electronic Information; 13–15 May 2011; Guangzhou, China. pp. 395-397.