Control Engineering Practice 18 (2010) 733–746

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Hardware and software architecture for state estimation on an experimental low-cost small-scaled helicopter

P.-J. Bristeau [a,*], E. Dorveaux [a], D. Vissière [b], N. Petit [a]

[a] Centre Automatique et Systèmes, Unité Mathématiques et Systèmes, MINES ParisTech, 60, boulevard Saint-Michel, 75272 Paris, France
[b] SYSNAV, Zone industrielle B, 1, rue Jean de Becker, BP 86, 27940 Aubevoye, France

ARTICLE INFO

Article history:
Received 1 April 2009
Accepted 25 February 2010
Available online 31 March 2010

Keywords:
Low-cost sensors
Embedded systems
MEMS
Unmanned aerial vehicle
Autonomous helicopter
Data fusion

ABSTRACT

This paper reports the design and testing of an embedded system for a low-cost small scaled helicopter (Benzin Acrobatic from Vario^{TM} with a 1.8 m diameter rotor). The sensors under consideration are an Inertial Measurement Unit (IMU), a GPS, a magnetometer, a barometer and on–off switches serving as take-off and landing detector. Along with one PC board and one micro-controller, they represent a total cost below USD 3000. By contrast to other experiments reported in the literature, the presented work do not rely on any accurate IMU or GPS systems which costs are, separately, largely above the mentioned amount of USD 3000. To compensate the weaknesses of this low cost equipment, efforts focus on designing a robust, dependable and sufficiently embedded system, which exploits an accurate flight dynamics model. This improves the prediction capabilities of an embedded extended Kalman filter that serves for data fusion. The main contribution of this paper is to detail, at the light of a successful reported autonomous hovering flight, the hardware, software architectures and the derivation of the model along with its calibration. Numerous implementation details are presented and the relevance of some modeling hypothesis is discussed.

1. Introduction

Numerous military and civilian applications require the use of unmanned aerial vehicles (UAV) with various degrees of autonomy (see Wise, 2004). Prime examples are borders surveillance, ground attacks, forest fires monitoring, reconnaissance, or civil engineering tasks. The always increasing performance of micro-electro-mechanical systems (MEMS) inertial measurement unit (IMU, device composed of a trihedron of gyrometers and a trihedron of accelerometers (see Titterton & Weston, 2004)), and low cost GPS have given them roles of enabling technologies for such applications. Yet, challenges are numerous. In particular, down-scaling of helicopters has appeared very difficult, while these aerial platforms represent a very good choice for outdoor applications under moderate to large wind gust disturbances while keeping the range of action relatively large. In this field, several teams have provided answers to certain technology questions related, in particular, to autonomous flight. Among these are the outstanding experiments conducted by Mettler (2003), Mettler, Tischler, and Kanade (2001, 2000a, 2000b), Abbeel, Coates, and Quigley (2007) and Ng et al. (2004). A R-max helicopter (with a 3.5 m rotor) from Yamaha was used at Carnegie Mellon University, Mettler (2003). An X-cell was used at Massachusetts Institute of Technology (this model has a 1.7 m rotor) by Gavrilets, Shterenberg, Dahleh, and Feron (2000).

While accurate dynamics models for such helicopters have been known since the work of Mettler et al. (2000b) and Gavrilets, Mettler, and Feron (2001), in all these reported experiments, classic filtering methodology was used. In details, a very general 6 degrees of freedom (DOF) rigid-body model was considered and was reported to be sufficient in coordination with good quality sensors. The reason for this is that the sensors considered in these studies provide accurate enough information for navigation and control. An accurate model was only used for control purposes, which include design, tuning, and simulation.

It is believed that cost reduction will yield a substantial speed-up in the spread of UAVs among military and civilian communities. As can be seen from various reported experiments, sensors represent an important part of the total cost of small size UAVs. In this paper, the focus is on really low-cost sensors. These sensors belong to the same technology group as those found on the discussed successful experiments, but they are really low-end in their categories. This raises some feasibility questions. What are the minimum quality requirements that one must impose on sensors to obtain a navigation system capable of providing real-time information sufficient to feed a stabilizing feedback controller? Which category of sensors can be used for each type of UAV? Which sensor is critical?

[*] Corresponding author. Fax: +33 1 40 51 91 65.
E-mail addresses: pierre-jean.bristeau@mines-paristech.fr (P.-J. Bristeau), nicolas.petit@mines-paristech.fr (N. Petit).

0967-0661/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.conengprac.2010.02.014

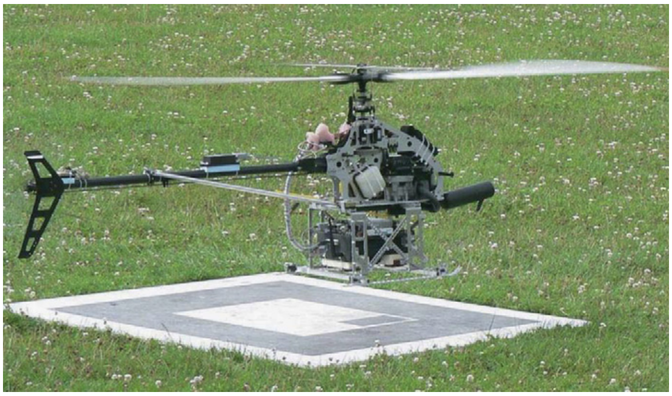

Fig. 1. "Goliath", the helicopter equipped with low-cost sensors in hover flight.

The answer is not solely contained in the sensors specifications considered individually and, most importantly, separately from a priori knowledge on the system dynamics. As it will be demonstrated, it seems that much improvement can be obtained from the knowledge of UAV models. Rather than considering data fusion algorithms based on a 6 DOF representation of the aerial vehicle under consideration, much attention should be paid to incorporating an accurate flight dynamics model into the data fusion algorithm.

Certainly, it is not possible to include all the equations found in the literature on helicopters. A complexity-accuracy trade-off must be made. Also, identification of the parameters appearing in the equations can be difficult. At the light of these points, some of the above-mentioned questions can be reformulated. What gain can be expected from including a dynamics model of the systems in the data fusion algorithms? Which are the physical phenomena that need to be accounted for and which ones can be neglected? These questions (among others in the same spirit) have been partially answered in the case of fixed wing aircraft. They will be addressed for small-scaled helicopters. As will appear, getting the maximum of low-cost sensors and flight dynamics models requires an efficient embedded system with a high refresh rate. This is why it was decided early in the presented project to design the considered embedded system.

The article is organized as follows. In Section 2, the hardware of the embedded system, including the sensors and the two communicating processing boards, is presented while Section 3 describes the software architecture and in particular the time stamping method. In Section 4, a description of the helicopter under consideration is made, discussion regarding interfacing and vibrational issues are conducted. Then a model for this aerial vehicle is presented and details about the neglected phenomena are given. In Section 5, the model is incorporated into a data fusion algorithm (extended Kalman filter). Coupling and ground effects are studied. By successively turning off each sensor, the robustness of the data fusion algorithm is tested and the beneficial effect of including the model in prediction phases of the extended Kalman filter is determined. Finally, Section 6 concludes and shows closed loop results obtained during an autonomous hovering (Fig. 1).

2. Embedded hardware

For the purpose of the presented research project, a versatile and simple real time embedded system was designed, which can be easily used as real time guidance and navigation system on various platforms such as small-scaled (typically less than 2 m wide) vertical take-off and landing aerial vehicles (VTOL as in Castillo, Lozano, & Dzul, 2005) or fixed wing aircraft, and ground vehicles with tank like dynamics (as in Morin & Samson, 2006; Vissière, Chang, & Petit, 2007a). Having such a versatile system is convenient to use in a fleet of heterogeneous vehicles asked to act cooperatively on the battlefield (see Kaminer, Yakimenko, Dobrokhodov, Lizaraga, & Pascoal, 2004; Olfati-Saber, 2006 for a discussion on operational scenarios).

The aerial vehicles represent a challenging applications in terms of navigation and guidance. The main reason for this is that these vehicles cannot easily go into any safe mode, as opposed to the ground vehicles which are, in comparison, slower and simpler. While it was proved that, with lowered performance expectations, it is possible to stabilize a fixed-wing unmanned air vehicles (UAV) by directly closing the loop with signals from well-chosen sensors (e.g. in Lee, Lee, Park, & Kee, 2003, the authors propose a solution to automatically control a fixed-wing UAV using only a single-antenna GPS receiver), it is considered by the vast majority of the UAV community that navigation systems require data fusion (see Cheng, Lu, Thomas, & Farrell, 2006). In facts, each sensor technology has its own flaws (among which are drift, noises, and possibly low resolution or low update frequency). Yet, large factors of accuracy can be gained by reconciliating their data.

Example of on-board data fusion applications are ubiquitous among autonomous vehicle control experiments. Reconciliating GPS and inertial measurement unit (IMU) measurements is a classic case-study. In Xiaokui and Jianping (2002), results of data fusion from a BeeLine GPS receiver from Novatel^{TM} and a miniaturized IMU are presented.

In Cremean et al. (2005), high-speed data fusion systems have been developed in view of the DARPA Grand Challenge. In this later experiment, several technological breakthroughs are presented using a high-end and powerful computer architecture. Software components communicate in a machine-independent fashion through a module management system.

The experiments cannot use such a high end setup, because the typical payload of the used aerial vehicles does not exceed 5 kg. Much smaller and lower-weighting systems can be considered though. In Jung and Tsiotras (2007), an embedded system is proposed which does not incorporate any powerful calculation board. A simple Rabbit Semiconductor RCM-3400^{TM} microcontroller is used to perform complementary filtering data fusion using a limited computational power. In the same spirit, in Jung et al. (2005), a low-cost test-bed for UAVs is presented. It is reported that the main advantage of designing such an autopilot from scratch is that, by contrast to commercially available products (see Cloud Cap Technologies, 2004; Micro Pilot, 2004), it provides full access to the internal control structures.

Here, a solution lying in the middle of the two previously mentioned categories of computational capabilities is presented.

2.1. System architecture

Early in the design process, one first constraint which appeared was the payload limitations of the considered flying machines. So the focus has to be on designing a relatively low weight embedded system.

A second issue that was also raised early in the design stage was that the real-time requirements of a control system for such small UAVs are very strong. This is mostly due to the short time horizons instabilities (which time-scale is typically of 40 ms). The control algorithm should be able to run with a period below 20 ms.

As previously mentioned, the sought-after system was meant to be versatile enough to be adapted to various platforms and control algorithms without significant change, and especially without raising new real-time scheduling issue.

Keeping these considerations in mind, it was decided to develop a robust two-processors embedded system (see Ben-Ari, 2006; Tanenbaum & Van Steen, 2002 for further insight on distributed systems), running two distinct softwares and communicating through a simple two-ways protocol. One processor is used to gather data from the sensors and to control the actuators. The other processor is used to perform the data fusion calculations (and possibly the control algorithms). It is fast enough to run a typical 15–30 dimensional states EKF algorithm with a low latency (to eventually produce satisfactory closed-loop results). The advantages of this structure are as follows: (i) task scheduling is easily programmed, because only one of the two processors is in charge of handling the numerous devices and I/O; (ii) new control algorithm are easy to load; (iii) depending on the computational requirements, the computation board can be easily upgraded without requiring any software changes or rising any concern about task scheduling; (iv) finally, the overall system is quite low-cost, since it relies on off-the-shelf components and can be easily maintained.

As exposed in Figs. 2 and 3, this (modular) embedded system is composed of a micro-controller, which is in charge of gathering information from all the sensors, and a calculation board. These two elements are connected by a serial interface. The micro-controller also has a downlink to a ground station and connections to the actuators and the remote control. Hardware components of the presented system are now detailed.

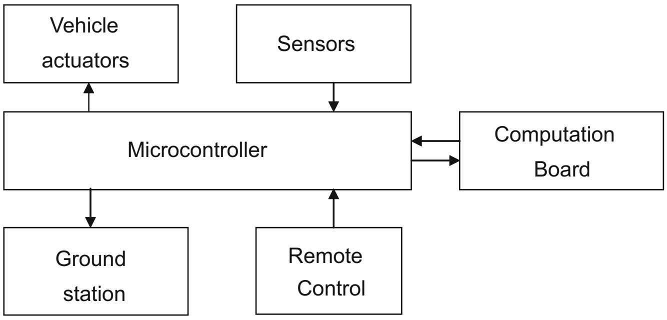

Fig. 2. Sensors and computation board connections to the central micro-controller.

2.2. Sensors

A series of useful sensors that needed to be incorporated into the embedded system was listed. In the context of the study, only low-cost sensors were considered. Among these are: an IMU, a GPS receiver, a pressure sensor, a magnetometer, and various switches. Other possibilities include LADARs (as used in Cremean et al., 2005), and sonars (as used in Vissière et al., 2007a), or cameras (as used in Hamel & Mahony, 2007). They are now detailed. In each case, the weight (in g), the cost (in USD), the dimensions (in mm), the update rate (f in Hz), and the protocol of communication (Comm.) are specified.

Inertial measurement unit (*IMU*): The IMU is a 3DM-GX1 from Microstrain^{TM}. It contains three angular rate gyroscopes, three orthogonal single-axis magnetometers, and three single-axis accelerometers, along with 16 bits A/D converters and a micro-controller. This IMU can deliver different messages, ranging from raw-data, to reconciliated measurements. In the presented setup, the IMU is asked to deliver only calibrated sensors data at a 75 Hz rate.

Weight	Cost	Size	f	Comm.
30	1450	39,54,18	75	RS232

Alternatively, this IMU could be replace by other models such as the Adis16405 from Analog Devices which is cheaper (approx. half prize). This last device which provides only raw data (and not reconciliated measurements) belong to the same class of sensors in terms of short term bias stability of the gyroscopes (which is the critical factor). Performance of the integrated accelerometers are also alike.

Global positioning system (*GPS*): The GPS is a TIM-LS from μblox^{TM}. Through a proprietary binary protocol, it provides position and velocity information at a 4 Hz rate. Position error is

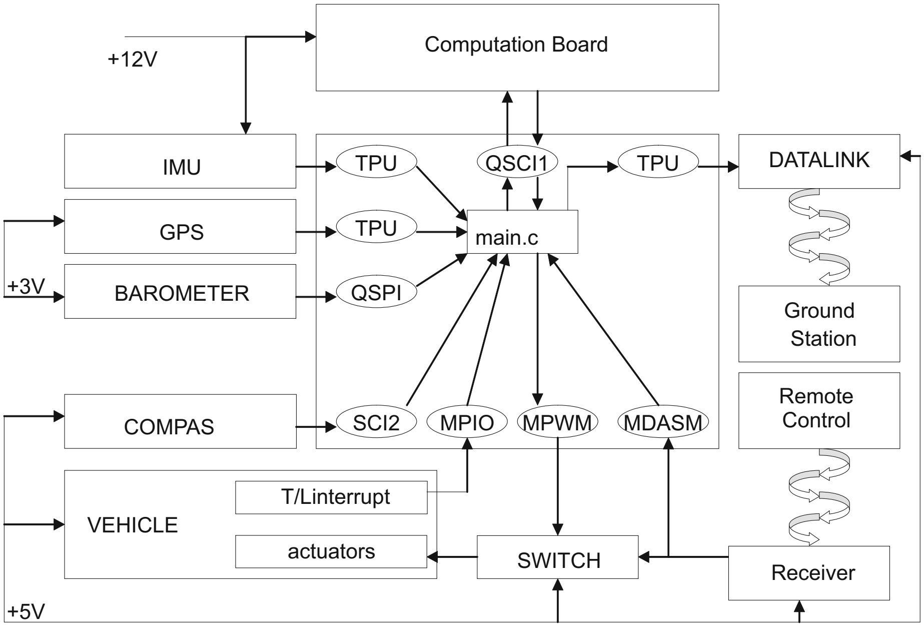

Fig. 3. Embedded system internal connections.

2.5 m (circular error probability, CEP) and velocity error is 2 m/s CEP.

Weight	Cost	Size	f	Comm.
23	100	32,47,9.5	4	RS232

The GPS receiver is not very tolerant against power supply voltage ripples. These can be kept below the 50 mV requirements thanks to a dedicated power supply regulator from TRACO$^{\text{TM}}$.

Barometer: The barometer is the MS-5534 from Intersema$^{\text{TM}}$. Using a SPI-type protocol, it gives calibrated digital pressure and temperature information. This device requires a 3 V power supply which is obtained through a fast response diode from the main 5 V power supply of the micro-controller.

Weight	Cost	Size	f	Comm.
2	14	5,4,2	20	SPI

Magnetometer: A HMR2300 three axis magnetometer from Honeywell$^{\text{TM}}$ was used. Its range is ± 2 G and it has a 70 μG resolution.

Weight	Cost	Size	f	Comm.
28	230	75,30,20	154	RS232

Take-off and landing detector: Being able to detect take-off and landing instants is necessary to properly initialize data fusion algorithms. In details, detection of the corresponding switches in the dynamics defines when the controls actually have an effect on the system. This is not the case when an UAV is on the ground. This detection is performed with on–off switches which can be located, e.g. on the landing gear. They deliver a logic signal which can be readily interpreted. To prevent electric arcs which might cause trouble to the connected micro-controller, a specific interfacing circuit was added. These switches can also be replaced by active switches which can be used to activate various devices such as digital cameras, or parachutes.

Weight	Cost	Size	f	Comm.
10	6	25,10,5	75	Boolean

The presented system is data-driven by the IMU. The main reason behind this choice is that the IMU is considered as a critical sensor.

2.3. MPC555 micro-controller

The micro-controller which serves as an interface for the sensors and actuators is a MPC555 Power PC from Motorola$^{\text{TM}}$. It runs a specific software developed especially for the presented system using the Phytec$^{\text{TM}}$ development kit. The reasons for choosing this micro-controller are as follows. This device provides a double precision floating-point unit (64 bit) which is convenient for potential embedded algorithms (even if this possibility is not used here since all computations are performed on the calculation board), it has a relatively fast 40 MHz clock, it has 32 bit architecture and 448 Kbyte of Flash memory and 4 MBytes of RAM. Most importantly, among the family of 32 bit kits, the MPC555 has substantial computational capabilities and a large number of versatile and programmable input/output ports. In particular, an extensive use of UARTs (QSCI1, queued serial communication interface and SCI2), TPUs (time process units used as customized additional UARTs), QSPI (queued SPI, MPIOs, modular I/O system and MDASM, MIOS double action submodule, see Motorola, 2000) is made. Finally, it is small (credit card format) and has a low weight.

Weight	Cost	Size	f	Comm.
25	450	72,57,8	All	All

No operating system is used on the micro-controller. Rather, the MPC555 runs a specific interrupts-driven software presented in Section 3. It gathers data from the sensors and send this information to a computation board for treatment. Once the message is received and validated, the calculation board carries out one navigation loop consisting of a prediction equation and an estimation equation of a extended Kalman filter. Details are given in Section 5. The commands are then sent to the actuators via the micro-controller.

2.4. PC computation board

The computing board is a PC running a customized Linux distribution. The PC board was selected among numerous models (mostly mini-ITX, and PC104) based on computational power, energy consumption, toughness, and price. A fan-less board was considered as the most relevant choice, due to the often observed failures of fans in mechanically disturbed environments.

The chosen fan-less calculation board has a standard mini-ITX PC architecture. Its processor is a 1.2 GHz C7-M from VIA$^{\text{TM}}$ designed for embedded applications. It can perform 1500 MIPS and has classic PC input/output ports such as a UART serial port (used as main data link with the micro-controller), an ethernet board (not used here), a VGA screen output (which can be used to monitor the system during debugging phases of the software and hardware development), a keyboard, and 4 USB ports (which can be considered for plugging future devices such has controllable cameras).

The operating system is installed on a bootable 1 Gbyte disk-on-chip system which prevents all possible mechanical failure associated to hard-drives. This flash memory device is directly connected to the IDE port of the mother-board. The board is powered by a pico-PSU$^{\text{TM}}$ power supply which provides various voltages ranging from 5 to 18 V. The computation software are written in C and can either be updated directly on the board via a `ssh` connection, or transferred, in a compiled form, from a remote PC. Custom scripts for compiling and distributing the executable code and configuration files are an efficient way to upgrade the software during development and testing.

Weight	Cost	Size	f	Comm.
800	350	170,180,40	1.5e6	RS232

3. Embedded software

In this section, the software used onboard the presented embedded system is. To run efficient data fusion algorithms, accurately time-stamped data are needed (as is stressed in Stengel, 1994). A method to achieve this for a collection of sensors is proposed here. This general technique was developed with a view to real-time applications in the field of position estimation by fusion of data from low-cost sensors (one can refer to Vissière, Martin, & Petit (2007b, 2007c) for more details on this type of applications). More precisely, delays between the physical measurements and the computation in the real-time algorithm must be reliably known and compensated for (most important is the variation of this delay depending on the sensor and over the time period the system is working). Otherwise, distortions and inconsistencies will unavoidably occur and introduce discrepancies, resulting in the failure or divergence of data fusion algorithms. The technique proposed addresses these issues.

In most *embedded positioning devices* (see e.g. Caccamo, Baker, Burns, Buttazzo, & Sha, 2005), information is repeatedly collected from the sensors at successive time steps. Data are 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, the technique presented below focus on a rather different weakness of implementation, which stems from the measurement system itself.

Consider a general purpose data fusion algorithm. It is generally assumed that all of the sensors can be interrogated simultaneously, and that they have trusted “upon request” behavior. In other words, it is usually assumed that they perform the measurement task exactly when they are asked to do so. This is not the case for most low-cost digital sensors. For instance, a careful investigation of the 3-axis magnetometer used in the present system revealed that it continuously performs periodic measurements, and delivers the latest one whenever a data request is made. A similar behavior can be observed on the IMU (Microstrain, 2006, pp. 9–10) and on most low-cost digital sensors. Unfortunately, this magnetometer does 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, except working directly with analog sensors. Secondly, it also must be assumed that information from all of the sensors can be read simultaneously. This task is usually performed by a micro-controller, 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 micro-controller 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. A reaction from the micro-controller at the exact time the information arrives is not achievable with the precision of the micro-controller clock because of timing jitter. The jitter depends on the interrupt model but is still present. However, a lower precision can be achieved which is already better than 1/10 of the sensor period (a few tenth of milliseconds instead of several milliseconds in the presented system). This is the method which is considered here and which is now detailed.

The micro-controller runs specific interrupt-driven software. Information from each sensor is transferred using a dedicated interrupt handler routine and stored along with a timestamp. More details can be found in Dorveaux, Vissière, Martin, and Petit (2009).

The data acquisition software running on the micro-controller is event-driven, driven by the IMU messages, which are 31 byte long. Once a message from the IMU has been received and validated by the micro-controller, the last available information from each other sensor and its associated timestamp is added in order to get a 116 byte message containing all of the onboard measurements and their associated timestamps. This message is then sent through a high-speed serial port. Fig. 4 describes the interruption part of the algorithm with more accuracy. It contains a schematic view of the interrupt management. Once the message is received by the ITX board, a loop of the data fusion algorithm detailed in Section 5 is run. Once the result is available, it is sent back to the micro-controller, which eventually uses it according to its own clock.

3.1. Solution method

Unlike the magnetometer, or the barometer, which do not deliver any timestamping information, the IMU, which can be considered as a regular clock, provides accurate timing information, included in its output message. The IMU is taken as a reference clock. Next, the process is organized in two steps. (i) Whenever a sensor (excluding the IMU) measurement has been received, the micro-controller measures the time elapsed from the latest IMU measurement (see again Fig. 4). (ii) This elapsed time is added to the IMU time and included in the gathered message.

These two steps are now detailed.

3.1.1. Timestamping at the micro-controller level

The reference clock is the IMU clock. The IMU tick count is included in its messages. The micro-controller clock is used to measure the elapsed time between an IMU message and the reception of measurements from the other sensors. The information from the other sensors is obtained under the form of messages received through various I/O ports (serial port, SPI port, etc.) using dedicated interrupt handler routines. To maximize performance, the task performed inside the interrupt handler routines is 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 Fig. 4, is exited as soon as possible, allowing other interrupts to come into play. Further tasks are done afterwards in the main loop. These include checking the validity of the message when a checksum is available, and converting the stored acquisition timestamps into timestamps relative to the IMU clock. Finally a single output message containing both sensors data and their timing information is formulated. The gathered message contains: (i) data collected from the various sensors. (ii) An absolute timestamp given by the IMU reference clock. (iii) A relative timestamp for every sensor data. This represents the time difference between the reception of the sensor measurements by the micro-controller and the acquisition of the data from the IMU contained in the same output message.

3.1.2. 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. This point is

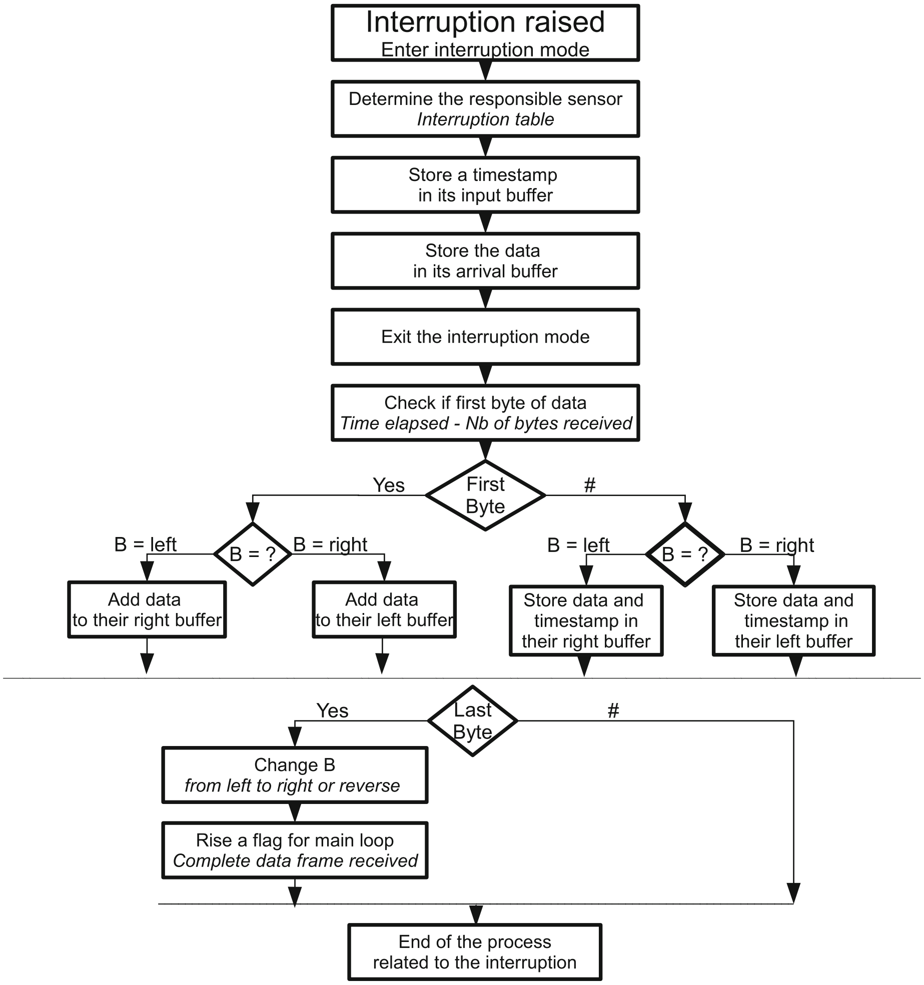
Interruption raised
Enter interruption mode
Determine the responsible sensor
Interruption table
Store a timestamp
in its input buffer
Store the data
in its arrival buffer
Exit the interruption mode
Check if first byte of data
Time elapsed - Nb of bytes received
First Byte
Yes
#
B = ?
B = left
B = right
Add data
to their right buffer
Add data
to their left buffer
B = ?
B = left
B = right
Store data and timestamp in their right buffer
Store data and timestamp in their left buffer
Last Byte
Yes
#
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. 4. 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. The boolean variable B indicates which buffer is in current use. 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.

supported by an experiment where the sensors were moved quickly from one steady state to another. Data sheets can provide an estimate of the absolute value of this delay, but large deviations are expected (at least for low-cost sensors). For the magnetometers, an offline identification easily yielded an estimate of this delay. The magnetometer responded in a few tenths of a millisecond, and the IMU reacted slightly later. The delay between the IMU and the magnetometer 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. Experimental small-scaled helicopter

In this section, the small-scaled helicopter which was equipped with the embedded systems presented earlier in the paper is described.

4.1. Aerial vehicle

A versatile hobby helicopter, the Benzin Acrobatic by Vario^{TM} was chosen because of its reliability (experiments never encountered any mechanical issue during more than 100 flights), its payload capacity (above 5 kg), and its simplicity of servicing. The rotor is 1.8 m wide, and the total mass of the helicopter is 7 kg (not counting the embedded system). The system is powered by a Zenoah^{TM} 23 cm^3 petrol engine which delivers 2000 W. Once fully loaded with the previously described embedded system (which weights approximately 1.8 kg), 20 cl of petrol approximately provide 20 min of flight.

4.2. Hardware interfacing and vibrational issues

Wiring the embedded system presented in Section 2 to the existing helicopter circuitry was achieved using some specific additional boards and connectors. During preliminary manual flights, it was useful to measure the pilot's orders in real-time. For that purpose, a 6 channels voltage follower circuit was used. Numerous LEDs were added to check the status of the system.

A central problem observed on-board helicopters is the 25 Hz vibrations induced by the main rotor blades. These vibrations generate a large amount of noise on the inertial sensors. In practice, these noises totally overwhelm the useful signals. Fortunately, it is possible to solve this issue by using well-chosen noise dampers. On the presented helicopter, it was decided that the micro-controller and the sensors would all be located on a board which would be physically connected to the frame of the helicopter through four spring–damper systems (see Fig. 5).

Experiments conducted on a vibrating table have shown that it was advantageous to keep the embedded system as compact and as rigid as possible. The total weight of the subsystem is about 600 g. Some of the batteries are attached to it to bring the weight close to 1.8 kg. This enabled the use of off-the-shelf dampers yielding appropriate cut-off frequencies. MV801-5CC dampers from Paulstra^{TM} were chosen for their ability to work with low masses vibrating at low frequencies. With these, the obtained vibration damping is satisfactory, with a cut-off frequency around 9 Hz. This is represented in Fig. 6. Further, resonant frequencies due to the engine frequency (around 160 Hz), the tail rotor frequency (around 115 Hz), and the tail boom were removed using a digital notch filter. The presented solution attenuates high frequency vibration inputs down to negligible levels.

4.3. Small-scaled helicopter modeling

In this section, the fundamentals of small-scaled helicopter modeling as used in the data fusion algorithm are recalled. Further details can be found in Mettler (2003).

4.3.1. Frame in use

A body reference fixed frame with origin at the center of gravity of the helicopter is considered. The x, y and z-axes coincide with the helicopter axis. These axes also coincide with the IMU inner sensors axis. In practice, to compensate for axis mismatch, an axis calibration procedure was used. In the following, subscript b refers to this body frame. As inertial reference frame, the north-east-down (NED) is considered. The X-axis is tangent to the geoid and is pointing to the north, the Z-axis is pointing to the center of the earth, and the Y-axis is tangent to the geoid and is pointing to the East. Subscript i refers to this inertial frame.

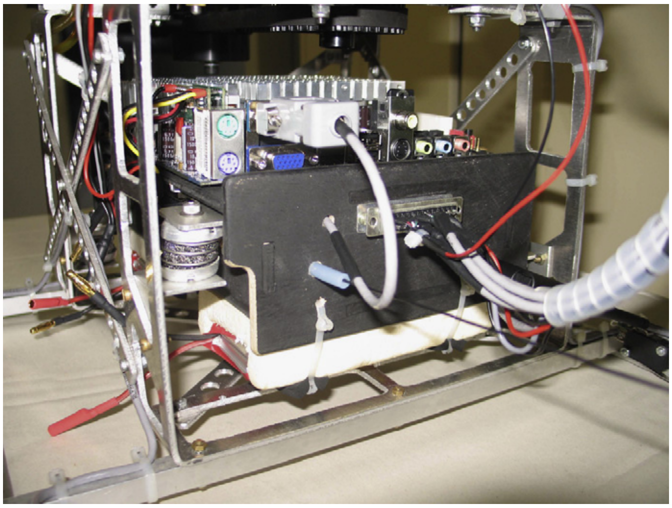

Fig. 5. The presented embedded system fitted into the (custom-built) landing gear of a small-scaled Vario Benzin helicopter. Springs and dampers are used to filter out vibrations from the main rotor blades.

4.3.2. 6 DOF rigid body dynamics

The rigid body considered is the helicopter without the rotor. From a dynamical system point of view, the state of the rigid body is described by the 12 following independent variables (Figs. 7 and 8)

- $X=[x\ y\ z]^T$ is the (vector) position of the center of gravity of the IMU expressed in inertial axes.
- $V=[u\ v\ w]^T$ is the (vector) velocity of the center of gravity of the IMU with respect to the inertial frame, expressed in body axes.
- $Q=[\phi\ \theta\ \psi]^T$ are the Euler rotation angles, i.e. the three successive rotations between the body frame and the inertial frame (even if quaternions are used for implementation, equations are easier to read with Euler angles. The reader can refer to Titterton & Weston, 2004 for changes of variables.).
- $\Omega=[p\ q\ r]^T$ are the turn rates of the body frame with respect to the inertial frame, expressed in body axes.

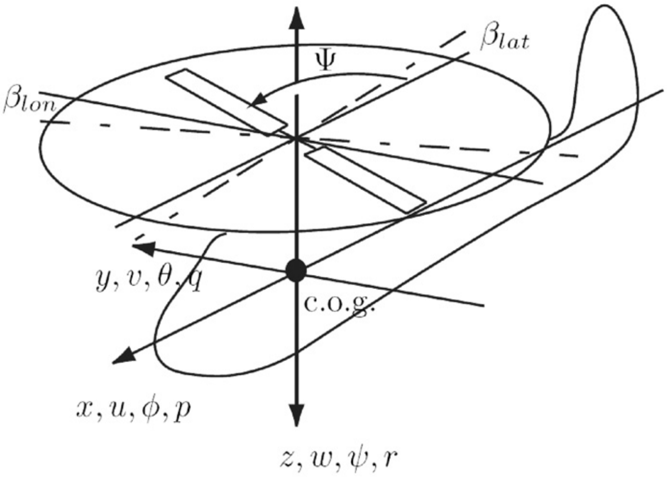

Fig. 7. Notations in the body frame.

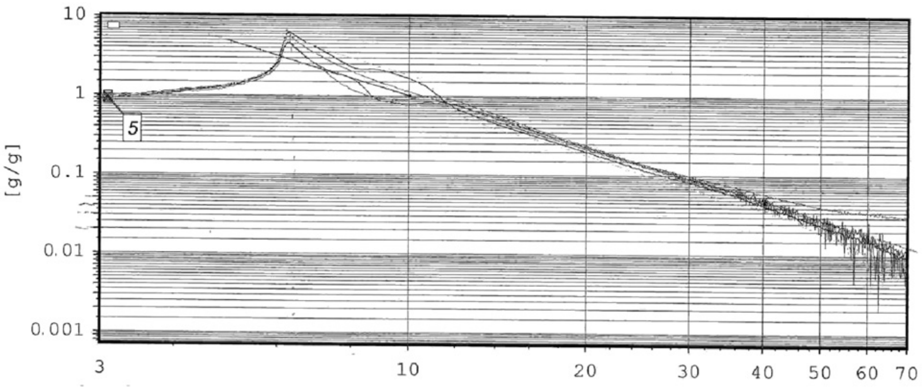

Fig. 6. Bode diagram showing the resonance peak and the cut-off frequency of the mechanical structure equipped with the sensors, the micro-controller, and the spring–dampers suspension. The various plots are obtained on varying locations on the vibrating structure and show a good spatial uniformity of the vibration damping.

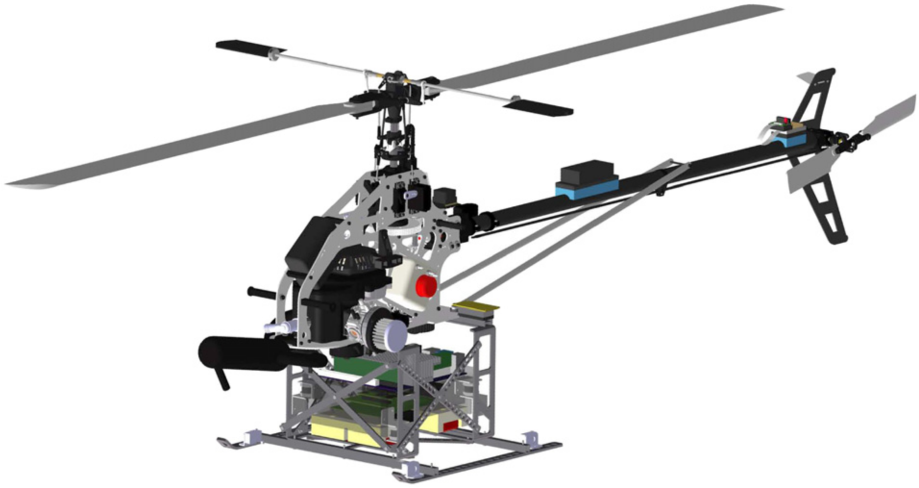

Fig. 8. Three dimensional view of the CATIA® model of the helicopter. Six hundred and eighty parts are modeled and kinematically linked together. The obtained model is pictured in Fig. 8. Payload, landing-gear, sensors, electronic devices, Li-Po batteries, and numerous add-ons are modeled to obtain accurate estimates of inertia matrix and position of the center of gravity.

The inputs are the forces F and the torques Γ. These inputs consist of the reaction of the rotor onto the body, the aerodynamic forces acting on the body, and the gravity forces. The mass of the helicopter is noted m, and its inertia matrix I, considered constant during the flight (i.e. neglecting fuel consumption in particular). With these notations, the dynamics of the 6 DOF rigid body are

$$\begin{cases} \dot{X} = R^T V \\ \dot{V} = -\Omega \times V + F/m \\ \dot{Q} = G(\Omega,Q) \\ I\dot{\Omega} = -\Omega \times I\Omega + \Gamma \end{cases} \tag{1}$$

with

$$G(\Omega,Q) = \begin{bmatrix} p + (q\sin(\phi) + r\cos(\phi))\tan(\theta) \\ q\cos(\phi) - r\sin(\phi) \\ (q\sin(\phi) + r\cos(\phi))\cos(\theta)^{-1} \end{bmatrix}$$

4.4. Rotor dynamics

One particularity of the presented helicopter is that a rigid rod connects the two blades of the main rotor. This features rules out any possibility of “coning effect”. This effect, commonly found on full-sized helicopters (see Padfield, 2007; Prouty, 2003), has an important impact on the thrust generated by the rotor. Because of the relative rigidity of the mentioned rod, it was decided to neglect this phenomenon. Yet, the vertical flapping β generates a torque M_β.

4.4.1. Main blades dynamics

According to the three hypothesis formulated in Mettler (2003)—(i) the vertical flapping angle is assumed small, (ii) the angle of attack of each blade is small, (iii) angular accelerations are negligible compared to the angular rate of turn multiplied by rotation speed—the main rotor blades flapping angles can be represented by two state variables a and b. These correspond to a first harmonic expansion of the periodic motion of the blades. These two states satisfy coupled linear first order dynamics driven by inputs corresponding to the control variables acting upon the rotor head:

$$\begin{cases} \tau_f \dot{a} + a = -\tau_f q + \dfrac{8}{\gamma\omega^2}\dfrac{k_\beta}{I_\beta} b + A_{lon}\delta_{lon}^{aug} \\ \tau_f \dot{b} + b = -\tau_f p - \dfrac{8}{\gamma\omega^2}\dfrac{k_\beta}{I_\beta} a + B_{lat}\delta_{lat}^{aug} \end{cases} \tag{2}$$

where γ is the Lock number, ω is the rotation speed of the rotor, k_β is the flapping hinge restraint spring constant, I_β is the blade moment of inertia about the flapping hinge and $\tau_f > 0$. A_{lon} is the longitudinal stick to cyclic pitch gearings, B_{lat} is the lateral stick to cyclic pitch gearings for the main bar. δ_{lon}^{aug} and δ_{lat}^{aug} are the inputs of cyclic pitch augmented by the Bell bar.

4.4.2. Bell bar dynamics

Following a similar approach, one can consider two additional states, c and d, corresponding to the Bell bar blades flapping angles. These satisfy a set of two uncoupled linear first order dynamics driven by inputs corresponding to the control variables acting upon the rotor head:

$$\begin{cases} \tau_s \dot{c} + c = -\tau_s q + C_{lon}\delta_{lon} \\ \tau_s \dot{d} + d = -\tau_s p + D_{lat}\delta_{lat} \end{cases} \tag{3}$$

where C_{lon} is the longitudinal stick to cyclic pitch gearings, D_{lat} is the lateral stick to cyclic pitch gearings for the Bell bar and $\tau_s > 0$. δ_{lon} and δ_{lat} are the inputs of cyclic pitch coming from the pilot. The Bell bar has an effect on the main blades dynamics due to the so-called “Bell mixer” mechanism. One can consider that the inputs in (2) are the real inputs δ_{lon} and δ_{lat} “augmented” as follows:

$$\begin{cases} \delta_{lat}^{aug} = \delta_{lat} + K_d d \\ \delta_{lon}^{aug} = \delta_{lon} + K_c c \end{cases} \tag{4}$$

Additionally, the Bell bar has an impact on the overall response time of the rotor. By adjusting its inertia (and aerodynamics) properties, this time constant can be rendered consistent with the skills of the pilot and its own response time in order to ease its task. This can help and make piloting easier, yet this is not a true stabilizing effect as could be erroneously understood.

4.4.3. Resulting forces and torques on the rigid body

The aerodynamics states a, b, c, d presented above can be used to define the orientation of the rotor thrust $\boldsymbol{R_{aero}}$ and the torque $\boldsymbol{M_{aero}}$ acting upon the rigid body of the helicopter. M_β represents the torque corresponding to the spring effect of the rotor hub. About steady state, these are

$$\boldsymbol{R_{aero}} = T[-a, b, -1]^T$$

$$\boldsymbol{M_{aero}} = hT[b, a, 0]^T$$

$$\boldsymbol{M_\beta} = k_\beta[b, a, 0]^T \tag{5}$$

where T is the norm of the thrust generated by the main rotor, and h is the height of the center of application of the thrust compared to the center of gravity of the helicopter. Around hovering $T \approx mg$. As will be discussed later on, the thrust is depending on the collective pitch control δ_{col} (and on z). More generally, the magnitude of the thrust cannot be easily computed. Quantitatively evaluating it requires computational fluid dynamics (CFD) solvers which are usually out of the scope of control-oriented studies. Rather, identification from flight data can be performed and they appear to provide satisfactory results.

Due to rotating parts, one should not forget the gyroscopic moment induced by the rotor on the rigid body. The non-negligible terms are a function of the rotation speed of the rotor ω and of the inertia of the rotor C_r compared to the axis of rotation, as follows:

$$\boldsymbol{M_{gyro}} = \omega C_r [q, -p, 0]^T \tag{6}$$

4.5. Tail rotor

To compensate the torque generated by the main rotor, the tail rotor produces a force acting with a large lever arm (0.95 m in this case). This result in a torque M_{tail} which depends on the tail rotor input δ_{ped}. This force also induces a lateral acceleration which is canceled out by a steady inclination of the main rotor. More complicated effects usually appear in practice. In particular, there exists some undesired interactions between the main rotor and the tail rotor through the air fluxes. These effects are always canceled by a low-level control system consisting of a rate-gyroscope located along the tail which measures r_m and acts upon the angle of attack of the tail rotor δ_{ped}. An identification of the response of this control system on the helicopter was performed. It is well represented by a 2 Hz low pass filter on the pilot's orders with a feedforward on the tail rate-gyroscope. Later, in filtering equations, it is considered that the input to the system is δ_{ped} the output signal of the tail gyroscope.

4.6. Weaknesses of the model

Before proceeding further and incorporating Eqs. (1)–(5) into a data fusion algorithm, it is important to comment on some of their apparent weaknesses.

4.6.1. Neglected phenomena

The model (1)–(5) presented above where the forces and torques are $\boldsymbol{R_{aero}}$ added to the gravity forces and the tail rotor thrust and $\boldsymbol{M_{aero}} + \boldsymbol{M_\beta} + \boldsymbol{M_{gyro}} + \boldsymbol{M_{tail}}$, respectively, is sufficient for state feedback control purposes. This point was raised by Mettler (2003), with extensive supportive experimental results. Yet, when the question is to estimate the attitude and position of the helicopter, it can be useful to account for further details. Among these are ground effect, actuators-induced lags and response time, and aerodynamics effect of the air flow from the main rotor passing by the body of the helicopter (rotor-induced body drag). These points will be developed in Section 5.2.

4.6.2. Parameter identification

Another point worth mentioning is that several key parameters must be available to obtain accurate estimates from the model (1)–(5). In particular, the inertia matrix I, and expression of the magnitude of the thrust T as a function of δ_{col} (see Eq. (5)) and the tail rotor torque M_{tail} as a function of δ_{ped}, response times of both the main rotor and the Bell bar (τ_f and τ_s in Eqs. (2) and (3)), as well as k_β, I_β, γ in (2), A_{lon}, B_{lat}, C_{lon}, D_{lat} and the position of the center of gravity are needed. In Section 5.2, explanation are giving how to identify these parameters.

5. Data fusion

This section is dedicated to the presentation of the data fusion algorithm which combines the measurements from the embedded systems presented in Sections 2 and 3, and the model detailed in Section 4.

5.1. Filter design and implementation

For state estimation, an extended Kalman filter which serves as a data fusion algorithm is used. The state of the filter has a total dimension of 23 in its current version. In details, 13 states were used to represent the 12 (independent) configuration states of the 6 DOF helicopter (quaternions are used to avoid well known singularities at $\theta = \pi/2$), four states are used to model the rotor dynamic (as discussed in Section 4.4), six states are used to model additional external (unknown) torques and forces. Theses forces and torques represent un-modeled terms, and, most importantly, wind gusts. A first approach can be to assume that these additional unknowns satisfy some first order dynamics driven by white noises ν_F (referring to forces) and ν_Γ (referring to torques), representing a colored dynamic noise. Dynamic noise variance must be taken large enough to capture the neglected dynamics, while the response time is tuned such that F and Γ are consistent with the dynamics while reducing the 25 Hz noise due to the main rotor rotation. More generally, performance certainly increases with the size of the state used for filtering (as long as added states are observable or stable), for example sensor drifts can be modeled this way. Limitations of the available computational power suggested to use only a small number of stated for these unknowns.

The measurement vector sent by the micro-controller to the computation board is composed of multi-rate data. The filter equations are presented below. In implementation, the covariance matrices can be initialized with values being consistent with the ranges of dynamics under consideration of the system. In details, typical speeds and accelerations are used. The filter updates are synchronized with the 75 Hz IMU measurements. Data time-stamps are used through interpolation functions. Classically, discrete-time update equations are considered. A special attention is paid to maintain the covariance matrices definite positive.

Note P_p the 23×23 covariance matrix of the state used for prediction, P_e the 23×23 covariance matrix of the state used for estimation, Q the 23×23 covariance matrix used in the noise dynamics, R_{IMU} the 7×7 covariance matrix considered in the sensor noise definition for IMU and barometer, R_{GPS} the 6×6 covariance matrix considered in the sensor noise definition for GPS position and velocity, X_p the 23 dimensional predicted state, X_e the 23 dimensional estimated state, A the 23×23 matrix of the system obtained by linearizing dynamics (1)–(5), C_{IMU} the 7×23 matrix and C_{GPS} the 6×23 matrix obtained by linearizing the measurement equation, $\dot{X} = F(X,U)$, and T_s the sample time (between measurements updates).

First, a prediction step is performed from time k to $k+1$, obtaining X_p and P_p. Then, the state is estimated through the measurements to obtain X_e and P_e. The updates are computed as follows from the multi-rate measurements. (They are done separately with distinct R_{XXX} and C_{XXX} as exposed in Stengel, 1994.)

$$X_p = X_e + F(X_e, U) T_s$$

$$P_p = (I + AT_s) P_e (I + AT_s)^T + QT_s + (AQ + QA^T) \frac{T_s^2}{2} + AQA^T \frac{T_s^3}{3}$$

$$Y_{pIMU} = [F - \vec{g}; \Omega; M]^T$$

$$Y_{pGPS} = [X; V]^T$$

$$K = P_p C_{XXX}^T (R_{XXX} + C_{XXX} P_p C_{XXX}^T)^- 1$$

$$X_e = X_p + K(Y - Y_{pXXX})$$

$$P_e = (I - KC_{XXX}) P_p (I - KC_{XXX})^T + KR_{XXX} K^T$$

where $_{XXX}$ corresponds to $_{GPS}$ or $_{IMU}$ depending on the current measurement. The measurement M stands for the earth magnetic field expressed in the body frame thanks to the predicted rotation angles. According to this, the code embedded into the calculation board is structured as follows:

1. *UART reading*: The UART driver gets the data sent by the micro-controller.
2. *Message decoding*: Once received, the data are transmitted to the user space. These data are composed of 8-byte word which must be decoded according to the sensor vendors proprietary protocols.
3. *Initialization*: All the values needed for state estimation are initialized. This also includes white dynamic and sensor noises, constants in use, and reference control values.
4. *Prediction*: In this step, the estimated state $\hat{X}_{e\_k}$ at time T_k is used to predict the state $\hat{X}_{p\_k+1}$ at time T_{k+1} accounting for the non-linear dynamics and the discrete sampling time T_s.
5. *Estimation*: In this step, the measurements and the predicted state $\hat{X}_{p\_k+1}$ at time T_k are used to update the estimated state $\hat{X}_{e\_k+1}$ at time T_{k+1}.
6. *Control*: New values of the control are computed with, e.g. a state feedback on the estimated state.
7. *Sending orders*: The UART driver is used (in interrupt mode) to transmit control values back to the micro-controller. In turn, the micro-controllers acts upon the servos.

The main loop detailed above is executed at a 75 Hz rate (approx. every 13.33 ms). Among the numerous tasks, the filter calculations take approximatively 6 ms, while receiving the data takes about 10 ms (mostly waiting time), and computing and sending back the control values takes 1 ms. An interrupt driven approach much be used to run all these task within the allowed 13.33 ms.

5.2. Details of the prediction step

In this subsection, the numerous details and phenomenon taken into account in the model are exposed. In particular, details about the identification of physical parameters, the effect of the collective pitch and the induced speed, the ground effect, the effect of the tail rotor pitch control, and the effect of the longitudinal and lateral cyclic pitch angles are given.

5.2.1. Identification of the inertia matrix and the center of gravity

As mentioned in Section 4.6.2, it is highly desirable, for sake of state estimation accuracy, to have good estimates of the inertia matrix and the position of the center of gravity of the helicopter. For that purpose, Dassault Systèmes's CATIA® software was used. Taking into account their various geometries and densities, 688 different parts were modeled and kinematically linked together. In particular, the basic structure, the engine, the blades, the payload, the landing-gear, the GPS, the power supplies, the Li-Po batteries, and numerous add-ons (such as flat cables or reservoir) were modeled. The obtained information is directly used to compute lever-arms of forces and to correct the IMU and GPS information. It should be noted that, while originally located under the rotor axis, the center of gravity is now 1.9 cm behind it.

5.2.2. Effect of collective pitch angle in vertical climb

The vertical displacement of the helicopter is directly impacted by the value of θ_0 which stands for the collective angle of attack of the blades (collective pitch angle). For small angles of attack, the magnitude of the generated thrust can be modeled as a linear function of the mean value of the angle of attack. Linearization around hovering equilibrium values yields

$$T = -mg - Z_{col}(\delta_{col} - Col_s) \tag{7}$$

where Z_{col} can be experimentally obtained as the ratio between the vertical acceleration and the control values δ_{col}, Col_s is the mean value of the control generated by the pilot for a stationary hovering flight in a flight zone where the ground effect is negligible. It can be noticed that $Z_{col} Col_s$ is greater than the gravity due to the compensation of Col_i which represents the reduction of the angle of attack due to the rotor induced velocity. In practice, the control δ_{col} is directly measured by the micro-controller which captures the signal sent to the servos.

5.2.3. Ground effect

Experimentally, substantial errors of the preceding model (7) are observed when the helicopter flies at low altitude. This phenomenon is a ground effect for which corrective terms are proposed in (7). Generally, a helicopter flying close to the ground requires less power than when it is flying far from it. Numerous models have been proposed in the literature for this phenomenon seen on various aircrafts. For aircraft, Rozhdestvensky (2000) proposes a ground effect theory appearing when the altitude is less than the chordwise length. In this approach, an additional thrust inversely proportional to the high is considered. Separately, Lefort and Hamann (1988) propose another point of view, and model the induced speed as being inversely proportional to the square of the distance between the rotor hub and the ground, when the altitude is less than one length of blade. Here, an exponentially decaying model is simply used which fits the experimental results well. This model is

$$T = -mg - Z_{col}\left(\delta_{col} - Col_s + Col_{GE} \exp\left(\frac{z}{Z_{GE}}\right)\right) \tag{8}$$

where Col_s equals the value of the control signal sent to maintain a stationary flight when the ground effect is negligible. Parameters Col_{GE} and Z_{GE} are positive. Their values were experimentally obtained during some long time hovering flights at various altitude. In Fig. 9, un-filtered signals which permit to estimate these parameters are reported. Accelerometer noises are at the birth of the large variance of the data. The fitted exponential law is depicted in black. Following results were obtained: $Z_{GE} = 0.5$ m which equals one main blade length, and $Col_{GE} = 180$, $Col_S = 610$. The additional acceleration due to ground effect is around $g/3$ on ground. Those value are consistent with the orders of magnitude of the helicopter.

5.2.4. Impact of the tail rotor pitch angle on the yaw motion

The tail rotor pitch angle δ_{ped} is the control signal sent to the tail rotor servo. Due to the affine relation between the actual control (PWM signal) and the obtained angle of attack (see Fig. 10), the resulting acceleration is of the form $Y_{ped}\delta_{ped}$, where $Y_{ped} > 0$. Using the experimentally obtained values of the acceleration of the helicopter along the y-axis, Y_{ped} is computed. Then, the resulting torque under the form $N_{ped}\delta_{ped}$ at the center of

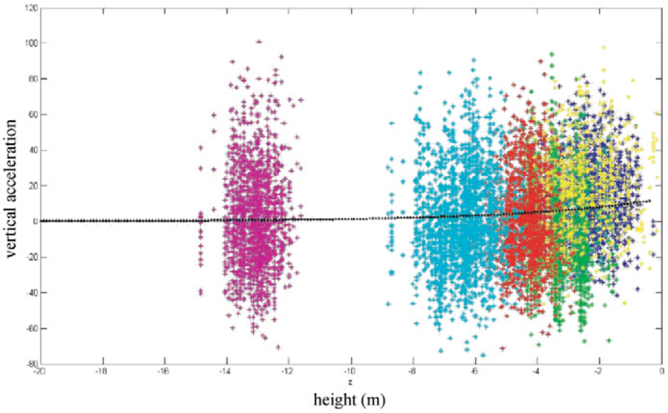

Fig. 9. Ground effect impact on the vertical acceleration.

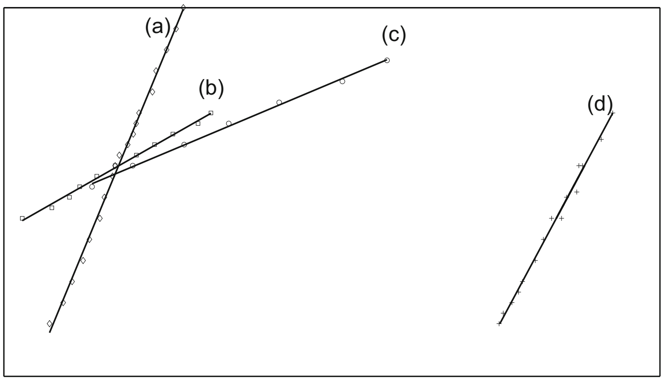

Fig. 10. Affine relation between (a) δ_{lon} and $\tilde{\theta}$, (b) δ_{lon} and θ, (c) δ_{col} and θ_0, (d) δ_{ped} and pitch angle for tail rotor.

gravity is evaluated by taking into account the mass and lever arm L_{ped}. This yields

$$N_{ped} = -L_{ped} m Y_{ped}$$

In the absence of yaw variation, δ_{ped} is such that the torque due to the tail rotor thrust and the torque generated by the main blade cancel each other. During a stationary flight, the mean value for δ_{ped}, noted δ_{peds}, provides with an estimate for the aerodynamic moment. The influence of the tail rotor on the helicopter dynamics is as follows. It generates a y-axis force F_{tail} and impacts on the rotational dynamics along the r-axis by a torque $M_{tail} = N_{ped}\delta_{ped}$. In details, one obtains

$$\begin{cases} F_{tail} = Y_{ped}\delta_{ped} \\ I_{ZZ}\dot{r} = N_{ped}(\delta_{ped} - \delta_{peds}) \end{cases}$$

where I_{ZZ} is the third diagonal term of the inertia matrix I. Modeling improvements could include the impact of a variation of collective pitch angle δ_{col} onto the aerodynamic moment (when δ_{col} increases, the aerodynamic moment on main blade also increases which explains why the engine throttle control is statically coupled to δ_{col} in order to keep a constant angular velocity for the rotor hub). This would imply that δ_{peds} is in fact a function of δ_{col}. Theoretically, this coupling could be identified online but it appears to be very difficult to estimate due to the high noise/signal ratio of the yaw gyroscope of the used IMU. Therefore, this phenomenon is neglected.

5.2.5. Effects of the longitudinal and lateral cyclic pitch angle onto the horizontal motion of the helicopter

Flapping dynamic: The dynamics of the rotor hub are as follows:

$$\begin{cases} \Theta(\Psi) = \Theta_0 - B_{lat}(\delta_{lat} + K_d d)\cos\Psi + A_{lon}(\delta_{lon} + K_c c)\sin\Psi \\ \tilde{\Theta}(\Psi) = \Theta_0 - D_{lat}\delta_{lat}\cos\Psi + C_{lon}\delta_{lon}\sin\Psi \end{cases} \tag{9}$$

where $\Theta(\Psi)$ (resp $\tilde{\Theta}(\Psi)$) is the blade pitch angle for the main bar (resp. for the Bell bar) which depends on Ψ the blade azimuth angle. Its formal harmonic decomposition, through the c and d variable in particular, can be used in identification experiments to derive K_c, K_d, A_{lon}, B_{lat}, C_{lon}, and D_{lat}. In Eq. (9) collective pitch angle θ_0 depends on δ_{col}. Interestingly, these experiments can be carried out under rest conditions, and need not in-flight data.

The static gains A_{lon}, B_{lat}, C_{lon} and D_{lat} play a role in the main rotor and the Bell bar. The control variables δ_{lon} and δ_{lat} are such that their zero values correspond to a null angle of attack of the blade (without taking into account collective angle of attack). The static gain C_{lon} (resp. D_{lat}) is the gain between cyclic controls and the pitch angle of Bell bar. It is directly measured by aligning the Bell bar with the lateral axis (resp. longitudinal axis), see Fig. 10(a). Similarly, A_{lon} (resp. B_{lat}) was experimentally obtained while keeping the Bell bar horizontal, see Fig. 10.

The amplifying effect of the Bell bar on control variables is characterized by the two gains $AC = A_{lon}K_c$ and $BD = B_{lat}K_d$. Those two gains were measured as the ratio between the angle of attack of the Bell bar and the main bar, for various values of c and d see Fig. 10. The rotor hub spring constant k_β was obtained by attaching masses on main blade and measuring the return torque as a function of flapping of the main bar. Flapping angles up to 8° were measured. The gain $AB = (8/\gamma\Omega^2)k_\beta/I_\beta$ may be computed through the value of I_β (obtained via CATIA).

Forces and torques balance: Using a detailed expression of $\boldsymbol{R_{aero}}$, Eq. (5) yields

$$\boldsymbol{R_{aero}} = \left(g + Z_{col}\left(\delta_{col} - Col_s + Col_{GE}\exp\frac{z}{Z_{GE}}\right)\right)\begin{bmatrix} -a \\ b \\ -1 \end{bmatrix} \tag{10}$$

A lever arm is used to compute the aerodynamic moment

$$\boldsymbol{M_{aero}} = M\left(g + Z_{col}\left(\delta_{col} - Col_s + Col_{GE}\exp\frac{z}{Z_{GE}}\right)\right)\begin{bmatrix} -hb \\ -ha \\ 0 \end{bmatrix} \tag{11}$$

The torque due to the rotor hub spring effect is given by

$$\boldsymbol{M_\beta} = k_\beta[b\ a\ 0]^T \tag{12}$$

5.3. Experimental estimation results

In this section, experimental state estimation results obtained onboard the presented helicopter are given. The robustness is studied by successively turning on and off every sensor.

5.3.1. Robustness, accuracy of the model, and sensors failure

The GPS velocity and position information, and the gyrometer angular rate of turn signal are successively turned off.

- *Loss of GPS*: To simulate a GPS loss during a flight, and to evaluate the relevance of the linear dynamic model as a substitute, the 4 Hz measurement was turned off and the data fusion extended Kalman filter was run with only 0.1 Hz GPS measurement, see Figs. 11 and 12. The flight under consideration shows typical forward flight speeds. The value of Euler angle are slightly disturbed. Position and velocity are strongly updated when a new measure appears, but,

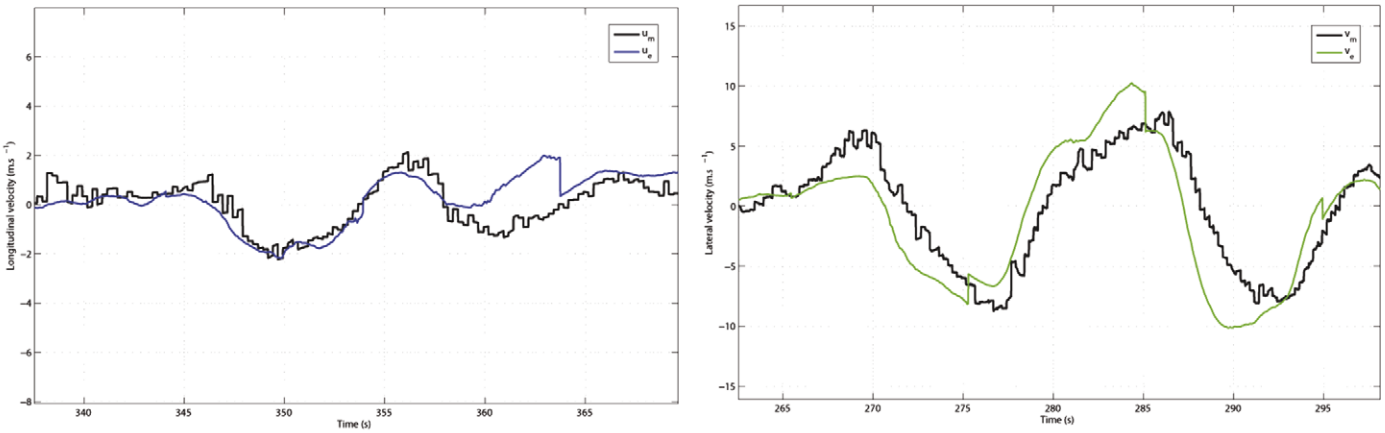

Fig. 11. Longitudinal velocity u (m/s) (blue) and lateral velocity v (m/s) (green) estimated and measured by GPS (black). Flight time (s) is reported on the x-axis.(For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

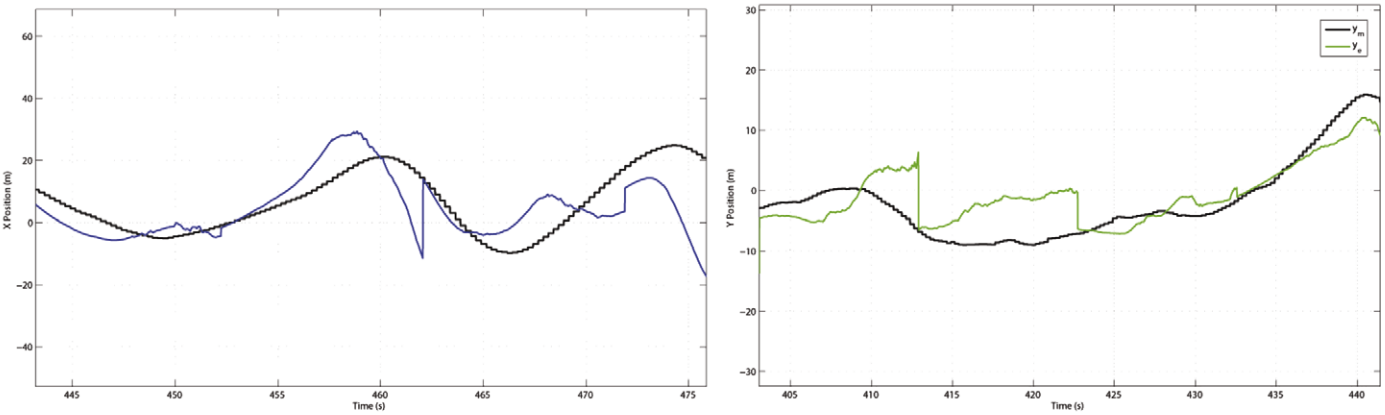

Fig. 12. Position: estimated along x-axis (blue) and y-axis (green), and measured by GPS (black). Flight time (s) is on the x-axis. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

interestingly, some parts of the flight are well predicted. In particular, the position estimates are relevant.

- *Gyrometer failure*: To test the relevance of the model with respect to the angular dynamics, the gyroscopes is turned off. Position and velocity errors remain large but do not grow unbounded, while angle estimates are debased. Yet, the predicted angular rate are really close to their real values, as can be seen in Fig. 13. In turn, the torques are well estimated. This point is consistent with the well predicted variations of the roll/pitch angles which can be seen in Fig. 13. These track the reference values obtained from the internal 6 DOF rigid-body state estimation algorithm IMU (which is not used otherwise).
- *Barometer failure*: To test the ground effect model, all sensors except barometer were turned on. Results are presented in Fig. 14. The estimated altitude relies on model mostly (vertical accelerometer is too noisy to expect to yield reasonable estimates, further the GPS accuracy is poor in the z-axis). The variance of altitude error is 3.5 m which is quite good. Altitude is always negative which is also good.

5.3.2. *Remaining weaknesses*

It appears that modeling of drag remains the source of non-negligible uncertainties. Also, masking effects between the two rotors seem a problem when the helicopter is moving.

6. Conclusion

In this paper, focus is put on incorporating important details in the flight dynamics used for data fusion onboard an experimental small-scaled helicopter. Estimation results prove the relevance of the approach which relies solely on low-cost sensors that are used close to their maximum potential thanks to a dedicated hardware and software architecture. Eventually, this real-time estimated state vector is used in a feedback controller. Classically, because of the substantial coupling terms in this multi-variable dynamics (see Balas, 2003), a LQR design was used in an approach similar to Gavrilets, Mettler, and Feron (2004). It is based on the Euler angles representation of the system. Slow/fast decomposition of the dynamics and knowledge of the pilot were used to sketch the values of the weighting matrices. This controller successfully stabilizes the helicopter in hovering flight. Typical results are presented in Fig. 15. During this 5 minutes autonomous flight, the position error remains within a three dimensional cylinder which is 1 m high and had a 3 m radius. This flight was obtained outdoor, under 20 km/h wind conditions. The variance of the error approximately equals 1 m in ground position, which is close to GPS error. In the vertical direction, the error variance is below 1 m vertically which is close to the absolute barometer error. For velocities, the variance of error is around 0.75 m s^{-1} horizontally, and 0.5 m s^{-1} vertically. Angular error remains within 3° in roll and pitch and 15° in heading. These results can be improved upon, especially by tuning the controller further. Most

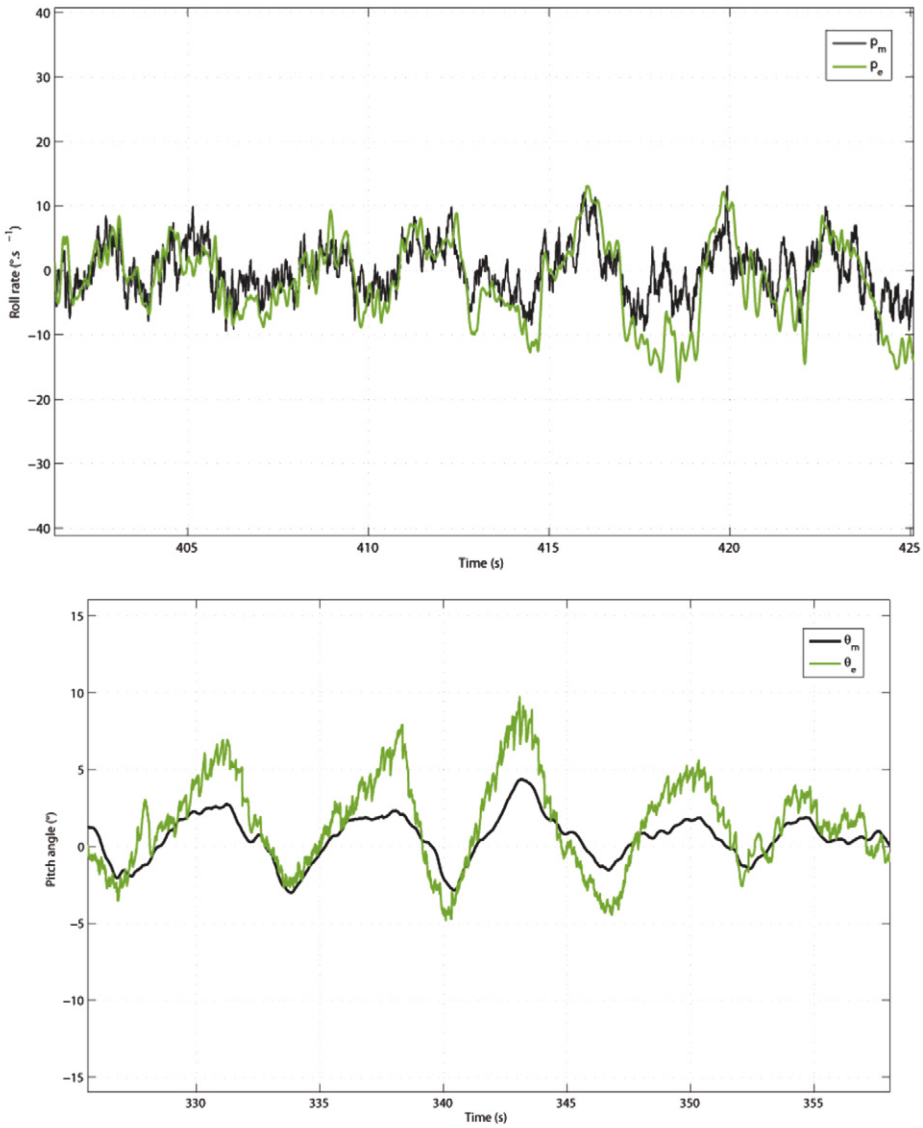

Fig. 13. Angular velocity p: (deg/s) estimated (green) and given from the IMU (black). Pitch angle θ (°): estimated (green) and given by the IMU filter (black). Flight time (s) is reported on the x-axis. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

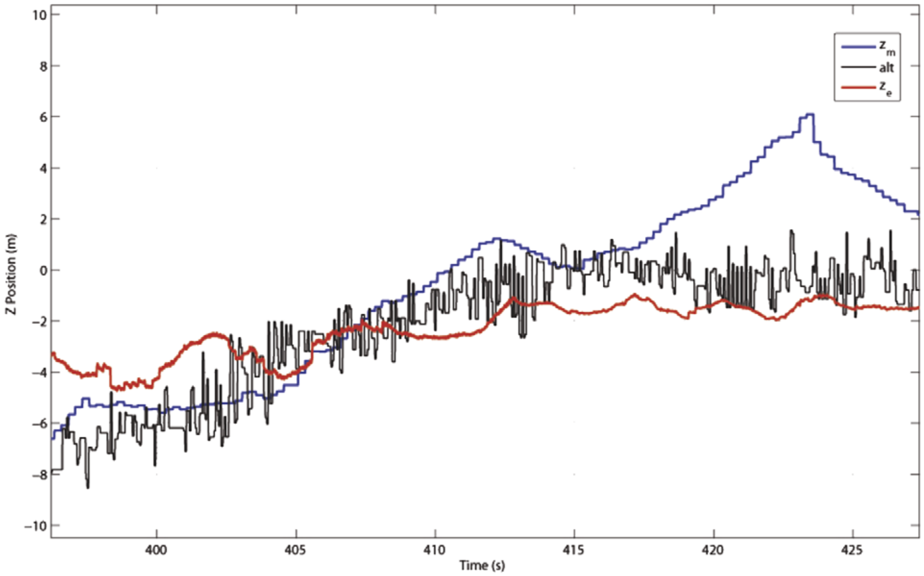

Fig. 14. Altitude: estimated (red) and measured by the barometer (black) and the GPS (blue). Flight time (s) is reported on x-axis.(For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

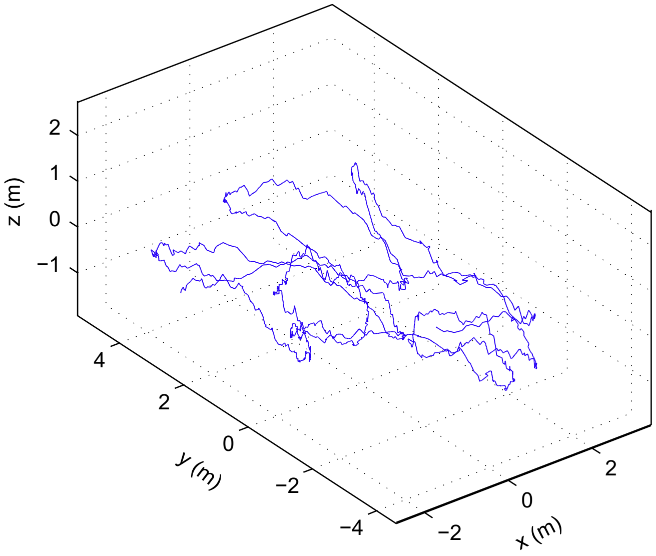

Fig. 15. Autonomous stationary flight. Position errors.

importantly, they represent a first step towards future developments which can incorporate forward motions, and trajectory following techniques (see e.g. Murray et al., 2003).

Acknowledgements

The authors wish to thank Gaël Desilles from DGA for his full technical and scientific support, and Johann Forgeard from DGA for his expert piloting skills.

References

Abbeel, P., Coates, A., Quigley, M., & Ng., A. Y. (2007). An application of reinforcement learning to aerobatic helicopter flight. In *Neural information processing systems* (*NIPS*) (Vol. 19).

Balas, G. J. (2003). Flight control law design: An industry perspective in fundamental issues in control. *European Journal of Control*, *9*, 207–226.

Ben-Ari, M. (2006). Principles of concurrent and distributed programming. In *Prentice-Hall international series in computer science* (2nd ed.). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

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

Castillo, P., Lozano, R., & Dzul, A. E. (2005). Modelling and control of mini-flying machines. In *Advances in industrial control*. London: Springer.

Cheng, J., Lu, Y., Thomas, E. R., & Farrell, J. A. (2006). Data fusion via Kalman filter: GPS and INS. In S. S. Ge, & F. L. Lewis (Eds.), *Autonomous mobile robots. Control engineering series*. London: Taylor & Francis.

Cloud Cap Technologies (2004). 〈http://www.cloudcaptech.com〉, Hood River, OR, USA.

Cremean, L. B., Foote, T. B., Gillula, J. H., Hines, G. H., Kogan, D., Kriechbaum, K. L., et al. (2005). Alice: An information-rich autonomous vehicle for high-speed desert navigation. *Journal of Field Robotics 9*.

Dorveaux, E., Vissière, D., Martin, A. P., & Petit, N. (2009). Time-stamping for an array of low-cost sensors. In 9th IFAC Workshop on Programmable Devices and Embedded Systems.

Gavrilets, V., Mettler, B., & Feron, E. (2001). Nonlinear model for a small-size acrobatic helicopter. In *AIAA guidance, navigation and control conference*.

Gavrilets, V., Mettler, B., & Feron, E. (2004). Human-inspired control logic for automated maneuvering of miniature helicopter. *Journal of Guidance, Control, and Dynamics*, *27*(5), 752–759.

Gavrilets, V., Shterenberg, A., Dahleh, M. A., & Feron, E. (2000). Avionics system for a small unmanned helicopter performing aggressive maneuvers. In *19th digital avionics systems conferences*.

Hamel, T., & Mahony, R. (2007). Image based visual servo-control for a class of aerial robotic systems. *Automatica*, *43*(11), 1975–1983.

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

Jung, D., Levy, E. J., Zhou, D., Fink, R., Moshe, J., Earl, A., et al. (2005). Design and development of a low-cost test-bed for undergraduate education in UAVs. In *Proceedings of the 44th IEEE conference on decision and control*.

Jung, D., & Tsiotras, P. (2007). Inertial attitude and position reference system development for a small UAV. In *AIAA Infotech at aerospace*.

Kaminer, I., Yakimenko, O., Dobrokhodov, V., Lizaraga, M., & Pascoal, A. (2004). Cooperative control of small UAVs for naval applications. In *Proceedings of the 43rd IEEE conference on decision and control*.

Lee, S., Lee, T., Park, S., & Kee, C. (2003). Flight test results of UAV automatic control using a single-antenna GPS receiver. In *AIAA guidance, navigation, and control conference and exhibit*.

Lefort, P., & Hamann, J. (1988). L'hélicoptère. Chiron, 2e edition.

Mettler, B. (2003). *Identification modeling and characteristics of miniature rotorcraft*. Boston: Kluwer Academic Publishers.

Mettler, B., Tischler, M. B., & Kanade, T. (2000a). Attitude control optimization for a small-scale unmanned helicopter. In *AIAA guidance, navigation and control conference*.

Mettler, B., Tischler, M. B., & Kanade, T. (2000b). *System identification of a model-scale helicopter*. Technical Report CMU-RI-TR-00-03, Carnegie Mellon University.

Mettler, B., Tischler, M. B., & Kanade, T. (2001). System identification modeling of a small-scale unmanned helicopter. *Journal of the American Helicopter Society*, 50–63.

Micro Pilot (2004). 〈http://www.micropilot.com〉, Stony Mountain, Canada.

Microstrain (2006). *3DMG-X1 data communications protocol*. Technical Report, Microstrain, March.

Morin, P., & Samson, C. (2006). Trajectory tracking for nonholonomic vehicles. In K. Kozlowski (Ed.), *Robot motion control: Recent developments*. London: Springer.

Motorola (2000). *MPC555/MPC556 user's manual*. User's manual, Motorola.

Murray, R. M., Hauser, J., Jadbabaie, A., Milam, M. B., Petit, N., & Dunbar, W. B., et al. (2003). Online control customization via optimization-based control. In T. Samad, & G. Balas (Eds.), *Software-enabled control, information technology for dynamical systems* (pp. 149–174). Hoboken, New Jersey: Wiley-Interscience.

Ng, A. Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., et al. (2004). Inverted autonomous helicopter flight via reinforcement learning. In *International symposium on experimental robotics*.

Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, *51*(3), 401–420.

Padfield, G. D. (2007). *Helicopter flight dynamics*. Oxford: Blackwell Publishing.

Prouty, R. W. (2003). *Helicopter performance, stability, and control*. Malabar: Krieger Publishing Company Inc.

Rozhdestvensky, K. V. (2000). *Aerodynamics of a lifting system in extreme ground effect* (1st ed.). Berlin, Heidelberg: Springer-Verlag.

Stengel, R. (1994). *Optimal control and estimation* (2nd ed.). New York: General Publishing Company, Ltd.

Tanenbaum, A. S., & Van Steen, M. (2002). *Distributed systems: Principles and paradigms*. Upper Saddle River, New Jersey: Prentice-Hall.

Titterton, D., & Weston, J. (2004). *Strapdown inertial navigation technology* (2nd ed.). The American Institute of Aeronautics and Aeronautics: Reston, USA.

Vissière, D., Chang, D. E., & Petit, N. (2007a). Experiments of trajectory generation and obstacle avoidance for a UGV. In *Proceedings of the 2007 American control conference*.

Vissière, D., Martin, A., & Petit, N. (2007b). Using distributed magnetometers to increase IMU-based velocity estimation in perturbed areas. *Decision and Control*, 4924–4931.

Vissière, D., Martin, A. P., & Petit, N. (2007c). Using magnetic disturbances to improve IMU-based position estimation. In Proceedings of the 9th European Control Conference.

Wise, K. A. (2004). Unmanned aircraft: The future in military aviation. In *Proceedings of the 6th IFAC World congress*.

Xiaokui, Y., & Jianping, Y. (2002). Study on low-cost GPS/DMU integrated navigation system. In *AIAA/AAS astrodynamics specialist conference and exhibit*.