An IMU/UWB/Vision-based Extended Kalman Filter for Mini-UAV Localization in Indoor Environment using 802.15.4a Wireless Sensor Network

Article *in* Journal of Intelligent and Robotic Systems · April 2013

DOI: 10.1007/s10846-012-9742-1

Alessandro Benini
University of Denver
16 PUBLICATIONS **150** CITATIONS
SEE PROFILE

Adriano Mancini
Università Politecnica delle Marche
136 PUBLICATIONS **982** CITATIONS
SEE PROFILE

S. Longhi
Università Politecnica delle Marche
407 PUBLICATIONS **4,007** CITATIONS
SEE PROFILE

J Intell Robot Syst (2013) 70:461–476
DOI 10.1007/s10846-012-9742-1

An IMU/UWB/Vision-based Extended Kalman Filter for Mini-UAV Localization in Indoor Environment using 802.15.4a Wireless Sensor Network

Alessandro Benini · Adriano Mancini · Sauro Longhi

Received: 29 June 2012 / Accepted: 13 July 2012 / Published online: 10 August 2012

Abstract Indoor localization of mobile agents using wireless technologies is becoming very important in military and civil applications. This paper introduces an approach for the indoor localization of a mini UAV based on Ultra-WideBand technology, low cost IMU and vision based sensors. In this work an Extended Kalman Filter (EKF) is introduced as a possible technique to improve the localization. The proposed approach allows to use a low-cost Inertial Measurement Unit (IMU) in the prediction step and the integration of vision-odometry for the detection of markers nearness the touchdown area. The ranging measurements allow to reduce the errors of inertial sensors due to the limited performance of accelerometers and gyros. The obtained results show that an accuracy of 10 cm can be achieved.

Keywords UAV · Indoor localization · Vision based odometry · EKF

1 Introduction

In last few years, Unmanned Aerial Vehicles (UAVs) have attracted attention in different fields, both civilian and military. They are able to perform tasks in hostile environments were access to humans is impossible or dangerous.

Precision navigation for these vehicles is a critical aspect that needs to be deeply analyzed.

In outdoor environments, UAV can localize itself exploiting different space-based satellite navigation systems, such as Global Positioning System (GPS). But the precision of satellite navigation systems is very limited, especially in civilian applications. The position accuracy that can be achieved with GPS is ≤ 1 m in military applications, and around 10 m in civilian applications. However using a technique known as differential GPS (D-GPS) or EGNOS corrections, in which a separate base receiver is fixed at a known point, civilian accuracy may be improved to 5 m. Although this is not as good as can be achieved using high frequency radar, it may still be adequate for some applications [1].

In order to improve the accuracy, different methods have been developed. Some of these improve the satellite accuracy by mean of data-fusion with inertial navigation systems (INS) [2, 3]. Others methods are based on vision [4, 5] or a combination of INS and Vision [6].

In GPS-denied and especially in indoor environments UAV localization can be successfully carried out exploiting combinations of wireless sensor networks, inertial navigation systems and vision-based odometry.

A. Benini (✉) · A. Mancini · S. Longhi
Dipartimento di Ingegneria dell'Informazione (DII),
Università Politecnica delle Marche, Ancona, Italy
e-mail: a.benini@univpm.it

Wireless sensor networks can be exploited successfully not only for communication between devices but also for localization purposes. In particular the IEEE 802.15.4a specifies two additional physical layers (PHYs): a PHY using ultra-wideband (UWB) and a PHY using chirp spread spectrum (CSS) both with a precision time-based ranging capability [7–10].

The UWB PHY is operating in three frequency bands: below 1 GHz, between 3 and 5 GHz, and between 6 and 10 GHz providing high ranging accuracy thanks to its large bandwidth. Furthermore, the UWB technology provides high data transfer, low power consumption, high spatial capacity of wireless data transmission and sophisticated usage of radio frequencies. UWB technology is based on sending and receiving carrier-less radio impulses using extremely accurate timing and it can be used in such applications where high bandwidth signals must be transmitted between devices.

The CSS PHY is operating in 2.4 GHz ISM band and does not support ranging features but the first 802.15.4a CSS chip (nanoLOC) developed by Nanotron has the ranging as additional (proprietary) function. It offers a unique solution for devices moving at high speeds because of its immunity to Doppler Effect and provides communication at longer ranges.

Concerning ranging measurements, in [11] a review of existing ranging techniques is provided. A RF Location System measures distances or angles between known points and an unknown position using several methods to obtain ranging measurement:

– Time of Arrival (ToA);
– Time Difference of Arrival (TDoA);
– Time of Flight (ToF);
– Angle of Arrival (AoA);
– Received Signal Strength Indication (RSSI).

An important drawback is that all these suffer from problems such as:

– Reflections;
– Multipath;
– Non-line-of-sight (NLOS) measurements: for communications, any signal path transfers useful information but for positioning only direct path signal can be used.

Concerning algorithms, the most used techniques are based on Kalman filter, especially on Extended Kalman Filter [12–14]. But, although the Extended Kalman filter is capable of providing real-time vehicle position updates, it is based on linear system models that might suffer from linearization when dealing with nonlinear models.

Other solutions are based on Unscented Transformation and Particle Filter. The main drawback for both is essentially the high computational load that usually prevents to use them in real-time applications. The UKF preserves the linear update structure of Kalman Filter. It uses only second order moments, which may not be sufficient for some nonlinear systems. In addition the number of sigma points used in UKF is small and may not represent adequately complicated distributions. Moreover Unscented Transformation of the sigma-points is computationally heavy and then not suitable for real-time aerial navigation applications [2].

This paper presents an extension of our previous work [15] based on the development of Localization Algorithm for Mobile Agents using Nanotron CSS System [16]. The limitation was the small bandwidth of the Nanotron CSS system (80 MHz) with a ranging accuracy in indoor environments of ± 2 m.

This paper studies the indoor localization of a small and low-cost UAV (AR.Drone from Parrot) using the UbiSense UWB Real-Time Localization System and low-cost IMU in conjunction with an Extended Kalman Filter (EKF) in order to improve the position accuracy.

The paper is organized as follows. In the next section a concise overview of the hardware/software used in this work is provided. Section 3 deals with the low-cost IMU characterization focusing the attention on the bias and scale factor problem. In Section 4 the kinematic model of the quadrotor is proposed. Then in Section 5 the EKF algorithm is analyzed while Section 6 deals with the improvement of the EKF with visual odometry. In the last section some experimental results are discussed.

2 Hardware/Software Setup

2.1 The UWB UbiSense Real-Time Localization System

The Ubisense Real-Time Localization System (RTLS) is a precision measuring instrument. Each sensor contains an array of 4 UWB antennas used for receiving the radio pulses from the tags (device to localize). The sensors perform ranging measurements using two different methodologies:

– TDoA;
– AoA.

This combination of methods provides a flexible system of localization in both indoor and outdoor environments, even in 3 dimensions.

The UbiSense location system is capable of 15 cm of accuracy (with 95 % of confidence level) and a maximum tag-sensor distance of 160 m. The tag update rate can vary from 10 updates/sec to 1 update/hour [17]. In addition the tag includes features for easy identification such as a motion detector sensor that activates the tag only when it moves [18]. The sensors are connected to a controller PC (in which the localization server engine runs) through a PoE (Power over Ethernet) switch as shown in figure (Fig. 2).

2.2 The Parrot AR.Drone

The AR.Drone (Fig. 1) is a small and low-cost quadcopter developed by Parrot. This quadrotor, currently available to the general public, is equipped with two cameras (one facing forward, the other horizontally downwards), a sonar height sensor and a flight-controller running proprietary software for communication and command handling based on Linux.

Commands and images are exchanged via a WiFi ad-hoc connection between the host machine and the AR.Drone.

The bottom camera is characterized by the following properties:

– 64 degree diagonal lens;
– Video frequency: 60 frames per second;
– Resolution: 176 × 144 pixels (QCIF).

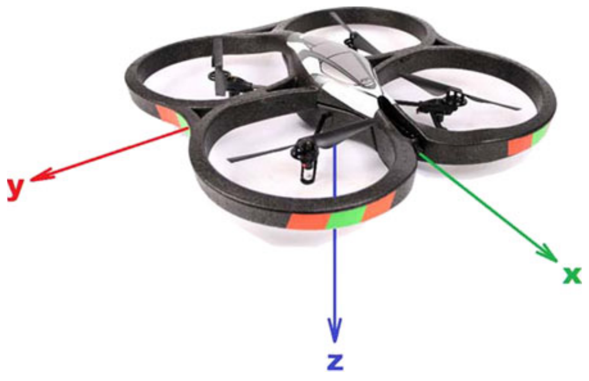


Fig. 1 The Parrot AR.Drone quadcopter with body frame axis orientation

The inertial measurement unit is composed by:

– A 3 axis digital MEMS (Bosh BMA150) accelerometer positioned at the center of gravity of the AR.Drone body. The accelerometer is used in a $+/-2$ g range and the acceleration measurements are acquired by an on-chip 10 bit ADC (see Table 1)
– A two axis MEMS gyroscope (InvenSense IDG500)
– A precise piezoelectric gyroscope (Epson XV3700) for the yaw angle and heading control.

Both gyroscopes measure up to 500°/s. These are analog sensors which outputs are acquired by the 12bits ADC and sent to the flight controller. Using particular settings the AR.Drone is able to send all the information provided by the sensors to the host PC. These data (called "navdata") take more bandwidth but contain also raw data from the ADC converter of the IMU.

Each set of data is identified by a *tag id* comprised between 0 and 20. By default only some data (identified by $tag = 0$) are provided in out-

Table 1 Bosh BMA 150 accelerometer specifications

Range	±2 g/±4 g/±8 g
Bias	±60 mg
Non-orthogonality	±2 %
Bandwidth	25–1500 Hz
Offset temperature drift	1 mg/K
Output noise	$0.5\ \text{mg}/\sqrt{\text{Hz}}$

put. Others data of interest in this work are identified by *tag* = 2 and *tag* = 3 which are defined as follow:

- **tag = 2** (NAVDATA_RAW_MEASURES_TAG) ADC output values from IMU called *raw_data*
- **tag = 3** (NAVDATA_PHYS_MEASURES_TAG): IMU data (calibrated with the internal calibration algorithm) called *phys_data*

2.3 The Hardware Setup

The hardware architecture is composed by 2 PC. The first PC hosts the UbiSense Location Engine and provides trilateration data on a UDP Server. The second PC, in which the drone control program runs, is connected to the first computer by Ethernet cable and to the AR.Drone through wireless network.

Ubisense Sensors and computers are connected together on the same local area network as reported in figure (Fig. 2).

2.4 The Software Setup

The software used in this work is mainly composed of three distinct modules:

- *Ranging UDP Server*: This application, developed using the UbiSense libraries, provides data from the Location Engine. At each time step the application generates a record containing trilateration data in double precision;
- *AR.Drone Control Software*: This module represents an extension of the AR.Drone C# SDK available in [19] improved with routines for:
  - Storing automatically frames from the bottom camera of the AR.Drone (this feature will be used in the vision module);
  - Retrieving raw data from the AR.Drone in real time (ADC output from the accelerometer and the gyroscope);
  - Sending commands to the drone using Joystick/Keyboard;
  - Combining trilateration data from Ranging UDP Server with the data provided by the AR.Drone in a unique data packet.

  This application is also responsible of sending commands to the drone and the execution delays in this software (that might origin from image elaboration) could interfere with the drone maneuverability. In order to avoid this drawback, the estimation of the drone position is performed by a dedicated Localization Algorithm module (LAM).
  From a point of view of data, this application can be seen as a data concentrator that provides, at each time step *k*, all the data to the LAM.
- *Localization Algorithm Module*: This module gets the data packet from the AR.Drone Control Software (NavData, frames and UbiSense trilateration) and elaborates them. As above mentioned, the communication between mod-

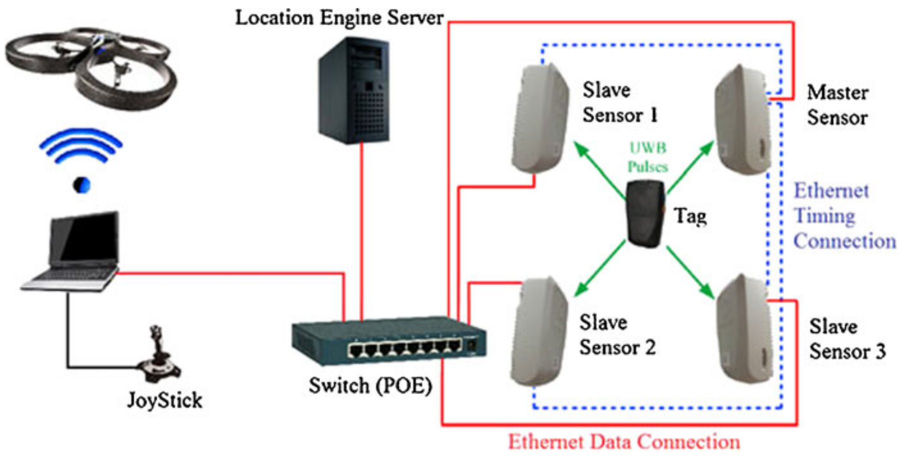


Fig. 2 The hardware architecture used during tests

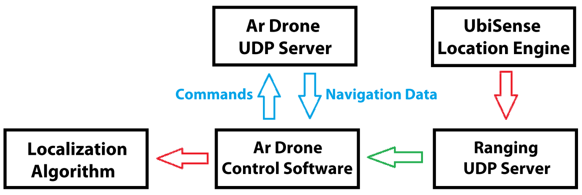


Fig. 3 The software architecture

ules using UDP sockets is useful if the data receiving is not a crucial factor (otherwise the TCP sockets should be used). In our application this is not a critical factor.

Figure 3 summarizes the Software Architecture.

2.5 Testing of Localization Algorithm in Simulation Environment

The localization algorithm developed in this work has been first tested in the new version of the 3D Simulation Environment developed in [20–22]. This Simulation Environment, based on SimplySim©SimplyCube [23] is a modular framework mainly oriented to the development and fast prototyping of cooperative Unmanned Aerial Vehicles. The framework combines the high realism of the simulation carried out in a three-dimensional virtual environment (in which the most important physic laws act) with the easiness of Simulink for fast prototyping of control systems. Furthermore it now includes modules for vision analysis and path-planning with State Flow Machines.

3 IMU Characterization

An IMU is generally composed by two orthogonal sensor triads. A triad consists of three mono-axial accelerometers, the other consists of three mono-axial gyroscopes. The two triads are nominally parallel and the origin of the gyroscope is defined as the origin of the accelerometer triad.

In recent years inertial sensors based on MEMS (Micro Electro-Mechanical Systems) technology have found applications in many different fields, thanks especially to their low cost and very small size [24]. But low cost sensor are generally characterized by poor performances. The most important drawbacks of a MEMS IMU are *bias*, *scale factor* and *cross-axis misalignements*.

In order to obtain good positioning accuracy it is necessary to deeply analyze the behavior of the sensors and realize special test calibrations, both in static and kinematic conditions.

3.1 The Accelerometer Mathematical Model

The acceleration along an axis can be expressed by the following relationship (Eq. 1), in which the thermal drift is not considered :

$$\ddot{z}_a = \ddot{z} + g + \epsilon_a + S_a g \tag{1}$$

where

- $\ddot{z}_a$ is the measured acceleration in output to the sensor;
- $\ddot{z}$ is the true value of the acceleration (at the considered point);
- g is the gravity acceleration;
- ϵ_a is the bias;
- S_a is the scale factor.

3.2 The Bias Estimation

The standard method used for calibrating IMUs was traditionally a mechanical platform rotating the IMU into different pre-defined orientations and angular rates. But these tests often require the use of specialized and expensive equipment.

A common not-expensive way to make an IMU calibration is the *six-position static test* [24, 25]. The six-position method requires the inertial system to be mounted on a leveled surface with each axis pointing alternately up and down. The bias can be calculated using the following equations:

$$\epsilon_a = \frac{\ddot{z}_{\text{down}} - \ddot{z}_{\text{up}}}{2} \tag{2}$$

where $\ddot{z}_{\text{down}}$ and $\ddot{z}_{\text{up}}$ are respectively two static measurements carried out holding the z axis of the accelerometer downward and upward.

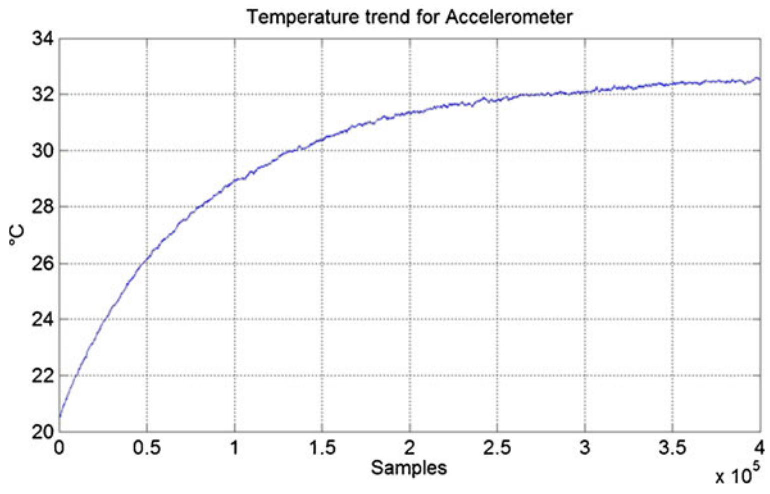


Fig. 4 Temperature trend for AR.Drone accelerometer

3.3 The Scale Factor Estimation

The scale factor can be described by the following relationship (Eq. 3):

$$S_a = \frac{\ddot{z}_{\text{down}} - \ddot{z}_{\text{up}} - 2g}{2g} \quad (3)$$

where g is the gravitational acceleration. A similar mechanism is applied in order to estimate the scale factor along the other two axis.

3.4 The Thermal Drift of the IMU

The IMU accelerometers and gyros are very sensitive to temperature as shown by Nebot and Durrant-Whyte [26]. As the temperature of the IMU changes, the associated bias and drift will change until the temperature reaches a steady value.

Considering the acceleration along the z axis in static condition the IMU temperature increases as an exponential law (see Fig. 4). The corresponding ADC output for z_{up} and z_{down} accelerations are shown in Figs. 5 and 6. As the temperature reaches a stationary value the ADC output stops to change. The bias and scale factor can be then calculated using stationary ADC output.

This is not critical in our application. Data are considered only when the thermal steady state of IMU is reached. After the heating of the sensor, the ϵ_a and S_a coefficients are supposed constant over a long period.

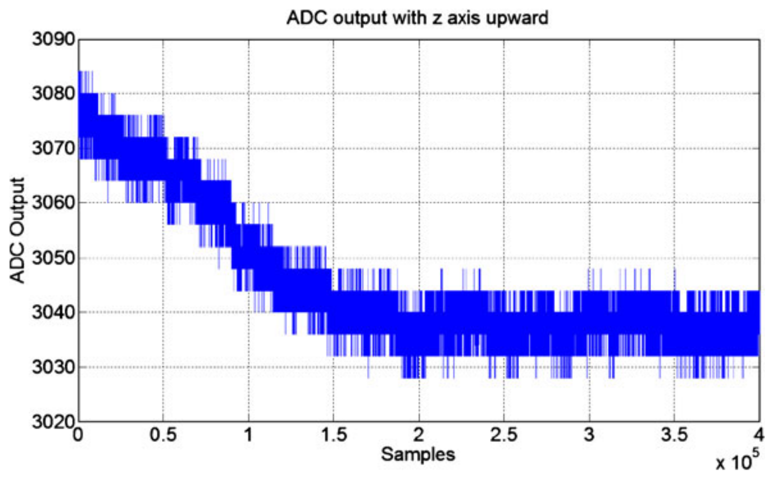


Fig. 5 ADC output with z axis upward

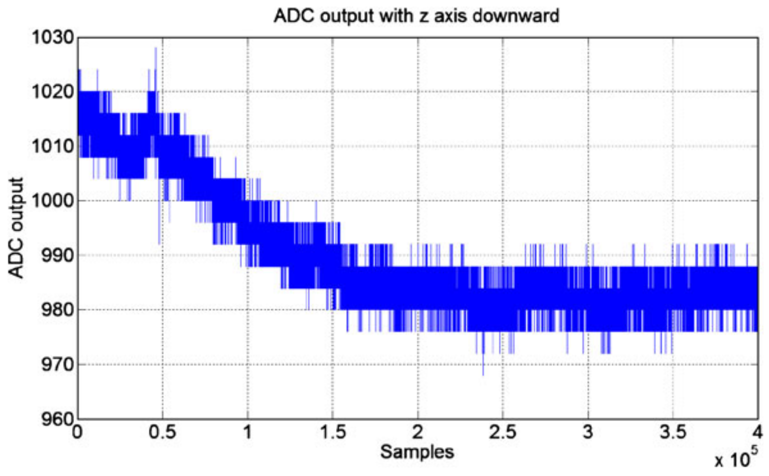


Fig. 6 ADC output with z axis downward

3.5 Comparison Between Calibrated Data and AR.Drone *phys_data*

In order to verify the calibration procedure described in the previous paragraphs, a *flight-test* has been carried out. The experiment consisted in collecting data from the AR.Drone during the hovering of the quadrotor. Because during take off the quadcopter can shows some drifts along x and y axis from the point of departure, data have been collected since the AR.Drone reached the default hovering altitude (1 m).

In Figs. 7 and 8 the red Gaussian describes the distribution of the acceleration values along x and y axis of the *phys_data* provided by the AR.Drone, while the blue dashed Gaussian describes the distribution of the acceleration values obtained with the procedure described above.

Table 2 summarizes mean values (μ) and standard deviations (σ) for each series of data, comparing data calibrated and data provided by the AR.Drone.

Figures 7 and 8 show that with the calibration algorithm the accelerometer performance is significantly improved (bias mean error on each axis <0.05 m/s^2).

4 The Kinematic Model of the Quadrotor

4.1 Axis Convention

To describe the motion of the UAV it is necessary to define a suitable coordinate system. For most problems dealing with aircraft motion, two coordinate systems are used. The first coordinate

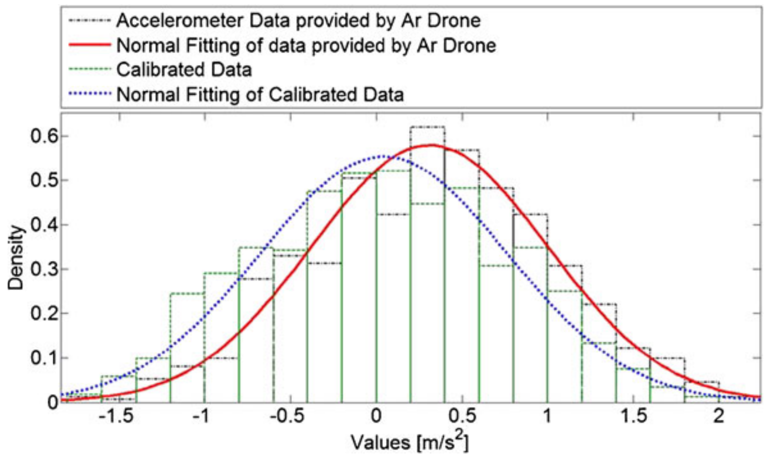


Fig. 7 Comparison between calibrated data and *phys_data*—x axis

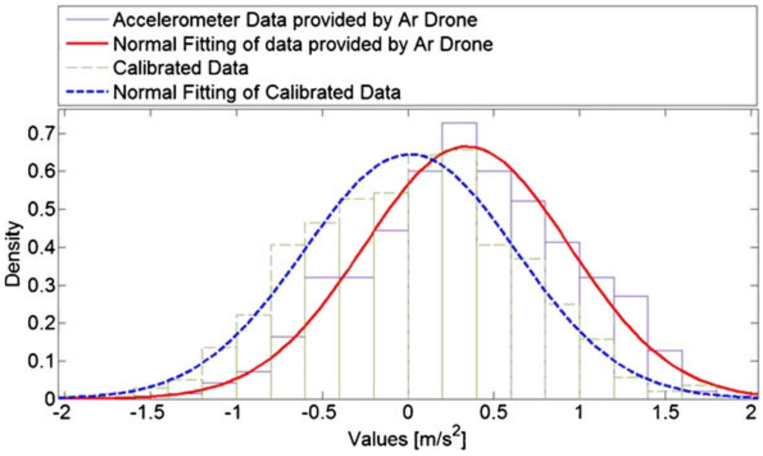


Fig. 8 Comparison between calibrated data and *phys_data*—y axis

system is fixed to the earth and may be considered for the purpose of localization. The second coordinate system is fixed to the UAV and is referred as a body coordinate system (in strapdown configuration).

In order to translate the acceleration from body frame to Navigation frame, the Direct Cosine Matrix is used (Eq. 4)

$$R_b^w = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{4}$$

where s represents sin, c represents cos and ϕ, θ, ψ, known as *Euler Angles*, are named *roll, pitch* and *yaw* (see Fig. 9).

Applying the rotation matrix R_b^w (Eq. 4), the accelerations on the world frame are (Eq. 5):

$$a_{\text{world}} = R_b^w a_{\text{body}} + G \tag{5}$$

Table 2 Comparison between acceleration values provided by AR.Drone and the calibrated acceleration values

Data	Mean μ	Std. σ
Acc x (*phys_data*)	0.3107	0.6884
Acc x (calibrated data)	0.0427	0.7203
Acc y (*phys_data*)	0.3425	0.6186
Acc y (calibrated data)	0.0114	0.6186

where G is the gravity vector in world frame, expressed as (Eq. 6):

$$G = [0 \quad 0 \quad -g]^T \tag{6}$$

and g is the gravity acceleration.

4.2 The Mathematical Model

Starting from the physical law that describes the uniform change of speed of a point p in one dimension:

$$p(t) = p(t-1) + \Delta p(t) = p(t-1) + \iint_{t-1}^{t} \alpha(\tau)\,\mathrm{d}\tau \tag{7}$$

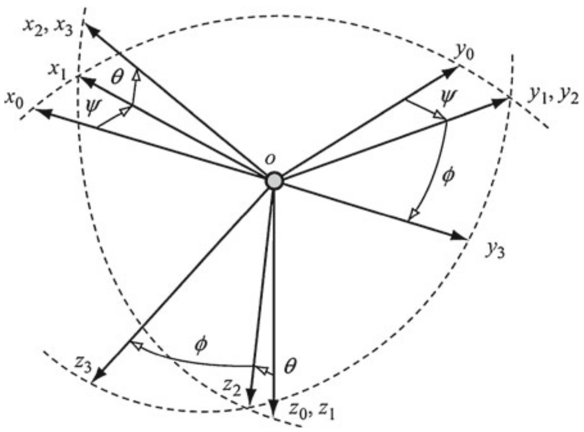


Fig. 9 Euler angles

the kinematic model used in the Kalman Filter at discrete time can be expressed as:

$$x_{k+1} = x_k + \Delta T v_{xk} + \frac{\Delta T^2}{2} a_{xk} \tag{8}$$

$$y_{k+1} = y_k + \Delta T v_{yk} + \frac{\Delta T^2}{2} a_{yk} \tag{9}$$

where $\{v_{xk}, v_{yx}\}$ and $\{a_{xk}, a_{yk}\}$ are respectively velocities and accelerations at step k.

5 The Extended Kalman Filter

The Kalman Filter is suitable for the estimation of x and y coordinates of a mobile device (tag) on the basis of ranging measurements made between the tag and at least three known points (anchors). Like the Kalman filter, the Extended Kalman Filter (EKF) is also carried out in two steps: prediction and estimation.

A fundamental issue with the EKF is that the distributions of the various random variables are no longer normal after undergoing their respective nonlinear transformations. The EKF is an *ad hoc* state estimation that only approximates the optimality of Bayes rule by linearization.

Let denote with $(\delta_{x,i}, \delta_{y,i})$ $(i = 1, ..., n)$ the x and y coordinates of the anchors and with $\mathbf{T} = (t_x, t_y)^T$ the unknown tag coordinates. The distance between an anchor and the tag is calculated in the following way:

$$d_i = \sqrt{(t_x - \delta_{x,i})^2 + (t_y - \delta_{y,i})^2} \tag{10}$$

The tag position can be obtained by trilateration as follows:

$$\mathbf{H} \cdot \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \mathbf{z} \tag{11}$$

where

$$\mathbf{H} = \begin{pmatrix} 2 \cdot a_{x,1} - 2 \cdot a_{x,2} & 2 \cdot a_{y,1} - 2 \cdot a_{y,2} \\ \vdots & \vdots \\ 2 \cdot a_{x,1} - 2 \cdot a_{x,n} & 2 \cdot a_{y,1} - 2 \cdot a_{y,n} \end{pmatrix} \tag{12}$$

and

$$\mathbf{z} = \begin{pmatrix} d_2^2 - d_1^2 + a_{x,1}^2 - a_{x,2}^2 + a_{y,1}^2 - a_{y,2}^2 \\ \vdots \\ d_n^2 - d_1^2 + a_{x,1}^2 - a_{x,n}^2 + a_{y,1}^2 - a_{y,n}^2 \end{pmatrix} \tag{13}$$

In the Extended Kalman Filter (EKF) the state transition and observation models need not be linear functions of the state but may instead be differentiable functions:

$$\begin{aligned} \tilde{\mathbf{x}}_{k+1} &= \mathbf{f}(\hat{\mathbf{x}}_k, \mathbf{u}_k, \mathbf{w}_k), \\ \tilde{\mathbf{y}}_{k+1} &= \mathbf{h}(\tilde{\mathbf{x}}_{k+1}, \mathbf{v}_{k+1}) \end{aligned} \tag{14}$$

where $\tilde{\mathbf{x}}_k$ and $\tilde{\mathbf{y}}_k$ denote respectively the approximated *a priori* state and observation and $\hat{\mathbf{x}}_k$ the *a posteriori* estimate of the previous step.

The state vector contains the predicted tag coordinates and velocities along x and y axis:

$$\mathbf{x}_k = \begin{pmatrix} x_k & y_k & v_{xk} & v_{yk} \end{pmatrix}^T \tag{15}$$

Referring to the state estimation, the process is characterized by the statistical variables $\mathbf{w}_k$ and $\mathbf{v}_k$ that represent respectively the process noise and measurement noise. $\mathbf{W}_k$ and $\mathbf{v}_k$ are supposed to be independent, white and normally distributed with given covariance matrix $\mathbf{Q}_k$ and $\mathbf{R}_k$. The observation vector $\boldsymbol{y}_k$ represents ranging measurements made between tag and anchors, and defines the entry parameter of the filter. Because in the analyzed system the predictor equation contains a linear relationship, the process function $\boldsymbol{f}$ can be expressed as a linear function:

$$\mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k + \mathbf{w}_k \tag{16}$$

where the transition matrix A is defined as follows:

$$A = \begin{pmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{17}$$

and T is the time sample.

The input control vector contains the linear acceleration ($\mathbf{u}_k$) of the quadcopter:

$$\mathbf{u}_k = \begin{pmatrix} u_{axk} \\ u_{ayk} \end{pmatrix} \tag{18}$$

Sensor measurements at time k are modeled in the Kalman Filter by the following equation (measurement model):

$$\mathbf{z}_k = H\mathbf{x}_k + \mathbf{v}_k \tag{19}$$

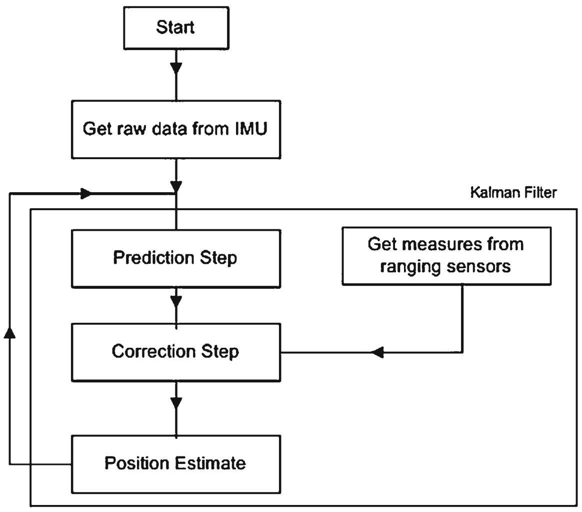


Fig. 10 Sensor fusion algorithm flow-chart

Because the UbiSense SDK provides only the (x,y,z) coordinate of the estimated position of the tag, the H matrix is:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{20}$$

The following flow chart shows how the sensor fusion algorithm works (Fig. 10):

6 Improving the Extended Kalman Filter with Vision-based Odometry

Today the vision-based odometry approach represents one of the most promising methodology to improve the accuracy and precision of unmanned aerial vehicles localization. An interesting review of methods for visual odometry is [27].

The approach here adopted is based on artificial well-known markers in terms of size and position in the considered environment. This kind of approach is suitable in the case of indoor localization. Feature based approaches based on SIFT/SURF and their variants are suggested for complex environments as the outdoor [28].

The first step (offline) is represented by the calibration of image sensor to derive intrinsic/extrinsic parameters of the camera. This step is performed using a standard chessboard pattern viewed from different point of view. The calibration is based on the Zhang approach [29].

In real-time the artificial marker are detected and geo-referred using the following approach:

- Get current frame;
- Convert from RGB to grayscale color space using the Principal Component Analysis to obtain high contrast image;
- Smoothing of image using a 3×3 Gaussian kernel;
- Adaptive thresholding;
- Polygon extraction;
- Shape Filtering (only polygons with four connected segments are considered);
- Extraction of perspective transform from 4 corners;
- Warp the image using the perspective transformation;
- Calculation of pose given a the set of extracted object points using the camera matrix and the

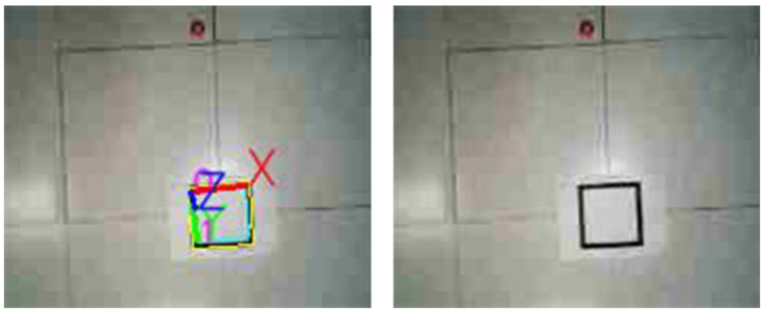


Fig. 11 An example of target detection using the bottom side AR.Drone webcam

Fig. 12 An example of QRCode used to improve the localization task in the considered indoor environment

distortion coefficient derived in the calibration stage;
– Decode if possible the content of marker if the QR-code is present.

In Fig. 11 an example of marker detection from bottom side AR.Drone camera is shown. The marker is square in shape with the side of 22 cm and 2 cm thick line. The idea is to use a set of well known marker optionally with coded information as shown in Fig. 12 using a QR-code.

The detection and recognition stages are performed online using a dedicated computer due to the constraints of computing unit installed on the AR.Drone. The time required to detect and recognize the marker is approximately 100 ms on a Intel Core 2 Duo machine running Windows 7 operating system. The extracted AR.drone pose referred to the detected marker is then used into the EKF to correct the estimation of position changing the weight of ranging and inertial data/measurements. The obtained results show that accuracy $\leq$0.1 m can be achieved. The bottom camera of AR.Drone was used due to future extension that focuses on the capability of take-off/landing from a mobile ground rover equipped with a well-known marker. This kind of camera is useful to improve these tasks which requires accuracy not available by the UWB sensors.

7 Experimental Results

In this section the results of some tests carried out in indoor environment are presented (Fig. 13). The reason of this choice is for the performances evaluation of the algorithm even in presence of multipath. The 2-D Cartesian coordinates of anchors have the following values (expressed in meters):

$$\begin{aligned}\mathbf{A}_1 &= [0.18 \quad 0.79]^T\\ \mathbf{A}_2 &= [1.79 \quad 8.11]^T\\ \mathbf{A}_3 &= [7.31 \quad 9.28]^T\\ \mathbf{A}_4 &= [4.62 \quad 0.23]^T\end{aligned} \tag{21}$$

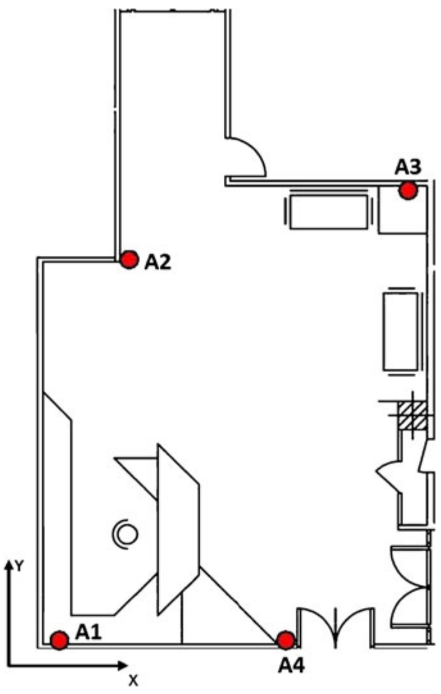


Fig. 13 Map of the indoor environment

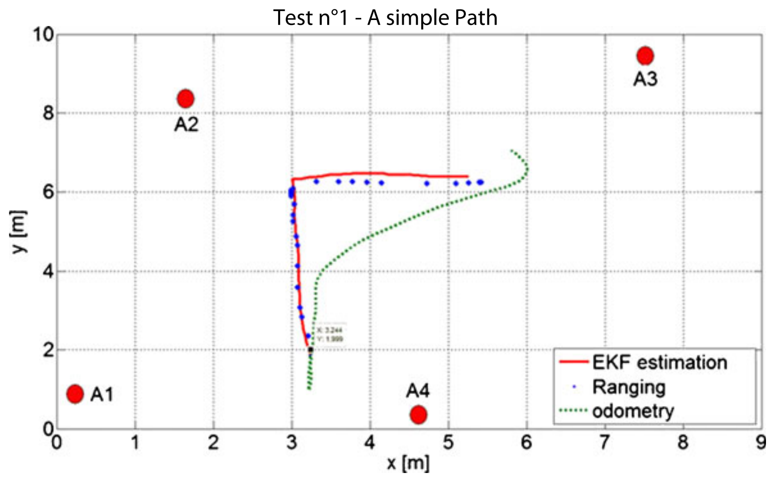


Fig. 14 Experimental result—test *n*°1

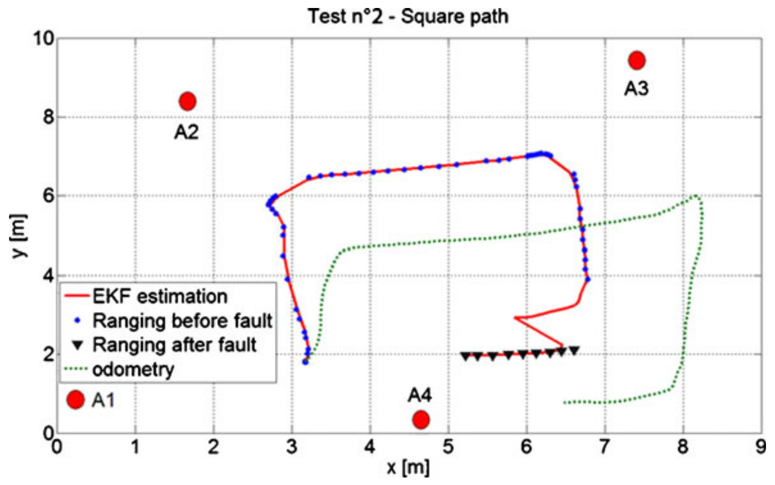


Fig. 15 Experimental result—test *n*°2

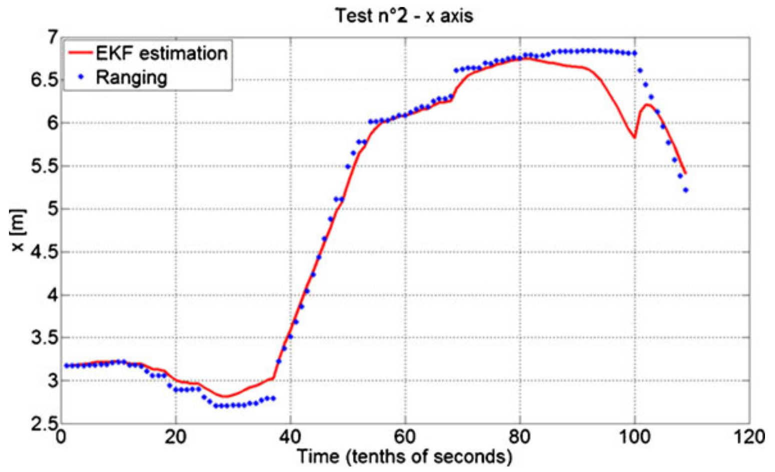
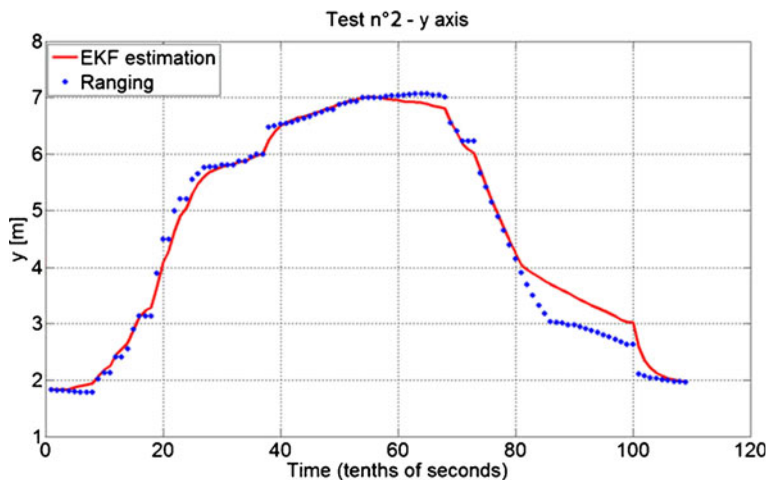


Fig. 16 Experimental result—test *n*°2—x axis

Fig. 17 Experimental result—test $n°2$—y axis



The initial position of the mobile agent is $[3.3 \quad 1.8]^T$ m. The speed of the UAV is set to the maximum AR.Drone velocity: 5 m/s.

The step time used in localization algorithm is 0.1 s in order to exploit the maximum update rate of the UbiSense Tag [17].

While the mobile agent moves along the path, the UWB tag node measures ranging from the four anchors and then a trilateration of (x,y) position is provided to the Localization Algorithm Module from the UbiSense Location Engine through the Ranging UDP Server.

7.1 Experimental Results without Vision-based Odometry

Figure 14 shows the position estimation (red line) using calibrated IMU data and ranging measurements (blue dots). As comparison, also the position estimation based on odometry (green dashed line) is reported.

7.2 Fault Simulation on Ranging Measurements

In Fig. 15 a more complex test has been carried out. The quadrotor is controlled to fly along a square path. During the flight a fault of the ranging sensors is simulated. In order to simulate the fault the localization algorithm is constrained to use the last available measure of trilateration provided by the Ranging UDP server through the AR.Drone Control Software. When the localization algorithm detects the fault (the same trilateration data over an interval T_f) it automatically increases the covariance matrix of the measurement process **R** and the error estimation covariance matrix **P**. In this way the quadcopter is constrained to localize itself using mainly odometric data. In Fig. 15 blue dots and black triangles represent respectively the ranging measurement before and after the fault.

Figures 16 and 17 show the x and y axis individually. The fault occurs at $T_f = [8 : 10]$ s. During the fault the quadcopter is able to localize itself with an error around 1 m. When the ranging system is re-established, the filter is able to correctly re-estimate the quadrotor position in less than 5 steps.

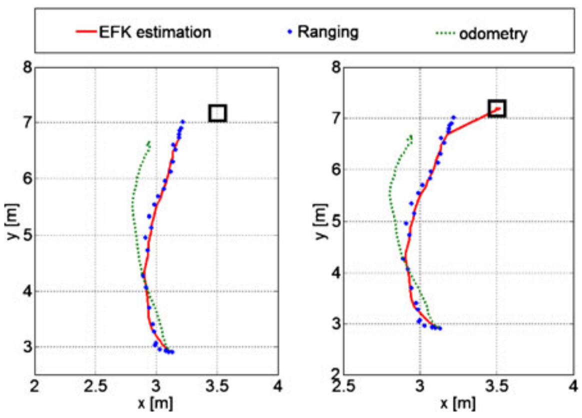


Fig. 18 Experimental result using also visual odometry

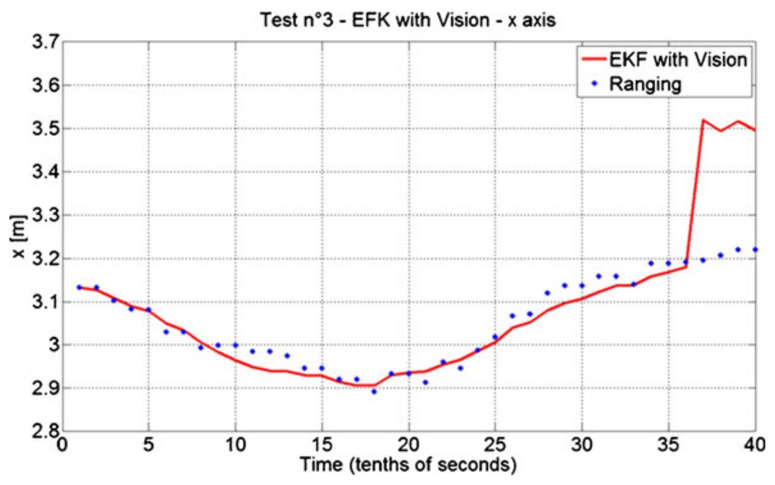


Fig. 19 Experimental result using also visual odometry—x axis

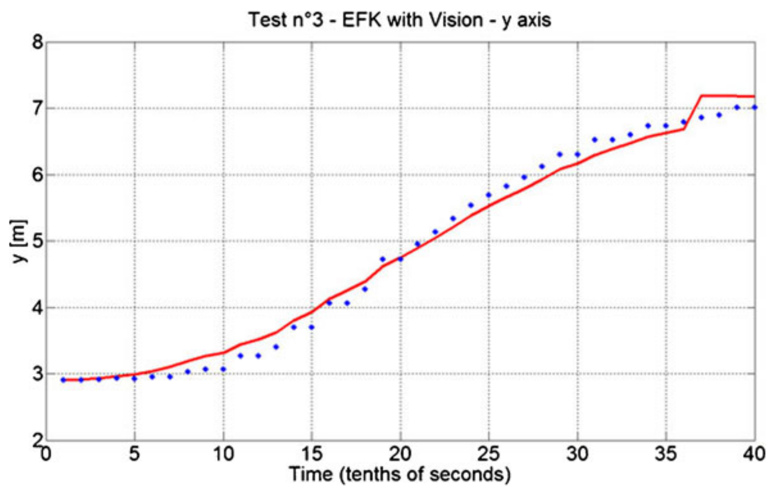


Fig. 20 Experimental result using also visual odometry—y axis

7.3 Experimental Results with Vision-based Odometry

In this section an experimental result of the quadrotor position estimation with vision-based odometry is reported. The marker is positioned at $(x, y) = [3.5 \quad 7.1]$ m. The initial position of the quadrotor is $(x, y) = [3.15 \quad 2.9]$ m. After take off, the quadrotor moves toward the marker. Figure 18 shows a comparison of the position estimation with and without the vision odometry.

If the marker is recognized by the bottom camera, the localization algorithm increases the weight of the vision odometry in the position estimation. At each frame, a boolean variable indicating the detection of the marker is associate. Figures 19 and 20 show the x and y axis individually.

When the quadrotor reaches the marker position, ($t = 3.5$ s), if the marker is detected, the boolean variable is set to *true* and the EKF reduces drastically the localization error, calculating at each time step the Δx and Δy position from the marker considering also the current body attitude.

8 Conclusion and Future Works

In this paper an Extended Kalman Filter for indoor localization using a 802.15.4a Wireless

Network, low-cost UAV and vision-based odometry was presented.

The calibration for MEMS is a critical aspect that needs to be carried out in order to improve the a priori state estimation. The calibration procedure provided an improvement of the MEMS sensor performances using a low-cost devices.

The obtained results reinforce the necessity of integrate additional sensors to obtain better results in terms of accuracy and precision.

In case of indoor environments the integration of a Laser Range Finder could significantly improve the overall performance.

The vision based system is based on the detection and recognition of artificial landmarks which can be installed in indoor environments but the approach can be easily extended to natural landmarks [28].

Future works will be steered to extend the set of sensors integrating visual information based on high-definition camera and to optimize the code to improve the overall performances. The integration of landing/take-off on/from a mobile ground platform will be also implemented combining ranging and visual data.

Acknowledgements This work is developed in the context of ARTEMIS-JU EU Project R3-COP. In particular the authors would like to thank Mauro Montanari and Riccardo Minutolo (Thales Italia S.p.A) for their support.

References

1. Barshan, B., Durrant-Whyte, H.F.: Inertial navigation systems for mobile robots. IEEE Trans. Robot. Autom. **11**(3), 328–342 (1995)
2. Nemra, A., Aouf, N.: Robust ins/gps sensor fusion for uav localization using sdre nonlinear filtering. IEEE Sens. J. **10**(4), 789–798 (2010)
3. Abdelkrim, N., Aouf, N., Tsourdos, A., White, B.: Robust nonlinear filtering for ins/gps uav localization. In: 2008 16th Mediterranean Conference on Control and Automation, pp. 695–702 (2008)
4. Sohn, S., Lee, B., Kim, J., Kee, C.: Vision-based real-time target localization for single-antenna gps-guided uav. IEEE Trans. Aerosp. Electron. Syst. **44**(4), 1391–1401 (2008)
5. Tisdale, J., Ryan, A., Kim, Z., Tornqvist, D., Hedrick, J.K.: A multiple uav system for vision-based search and localization. In: American Control Conference, pp. 1985–1990 (2008)
6. Rady, S., Kandil, A.A., Badreddin, E.: A hybrid localization approach for uav in gps denied areas. In: 2011 IEEE/SICE International Symposium on System Integration (SII), pp. 1269–1274 (2011)
7. Sahinoglu, Z., Gezici, S.: Ranging in the ieee 802.15.4a standard. In: Wireless and Microwave Technology Conference, 2006. WAMICON '06. IEEE Annual, pp. 1–5 (2006)
8. Rohrig, C., Muller, M.: Indoor location tracking in non-line-of-sight environments using a ieee 802.15.4a wireless network. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009, pp. 552–557 (2009)
9. Sikora, A., Groza, V.F.: Fields tests for ranging and localization with time-of-flight-measurements using chirp spread spectrum rf-devices. In: IEEE Instrumentation and Measurement Technology Conference Proceedings, 2007. IMTC 2007, pp. 1–6 (2007)
10. Krishnan, S., Sharma, P., Guoping, Z., Woon, O.H.: A uwb based localization system for indoor robot navigation. In: IEEE International Conference on Ultra-Wideband, 2007. ICUWB 2007, pp. 77–82 (2007)
11. Sayed, A.H., Tarighat, A., Khajehnouri, N.: Network-based wireless location: challenges faced in developing techniques for accurate wireless location information. IEEE Signal Process. Mag. **22**(4), 24–40 (2005)
12. Mao, G., Drake, S., Anderson, B.D.O.: Design of an extended kalman filter for uav localization. In: Information, Decision and Control, 2007. IDC '07, pp. 224–229 (2007)
13. Rullan-Lara, J., Salazar, S., Lozano, R.: Uav real-time location using a wireless sensor network. In: 2011 8th Workshop on Positioning Navigation and Communication (WPNC), pp. 18–23 (2011)
14. Purvis, K.B., Astrom, K.J., Khammash, M.: Estimation and optimal configurations for localization using cooperative uavs. IEEE Trans. Control Syst. Technol. **16**(5), 947–958 (2008)
15. Benini, A., Mancini, A., Frontoni, E., Zingaretti, P., Longhi, S.: Adaptive extended kalman filter for indoor/outdoor localization using a 802.15.4a wireless network. In: Proceedings of the 5th European Conference on Mobile Robots ECMR 2011, pp. 315–320 (2011)
16. Nanotron: Nanotron Technologies GmbH. http://www.nanotron.com. Accessed May 2012
17. Ward, A.: In-building location systems. In: The Institution of Engineering and Technology Seminar on Location Technologies, 2007, pp. 1–18 (2007)
18. Corrales, J.A., Candelas, F.A., Torres, F.: Hybrid tracking of human operators using imu/uwb data fusion by a kalman filter. In: Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction, HRI '08, pp. 193–200, ACM, New York, NY, USA (2008)
19. Hobley, S.: AR Drone C# SDK. http://www.stephenhobley.com/blog/2010/11/28/c-sdk-for-ar-drone-now-available/ (2010). Accessed December 2011
20. Benini, A., Mancini, A., Minutolo, R., Longhi, S., Montanari, M.: A modular framework for fast pro-

totyping of cooperative unmanned aerial vehicle. J. Intell. Robot. Syst. **65**, 507–520 (2012). doi:10.1007/s10846-011-9577-1
21. Mancini, A., Benini, A., Frontoni, E., Zingaretti, P., Longhi, S.: Coalition formation for unmanned quadrotors. In: Proceedings of the 7th International ASME/IEEE Conference on Mechatronics & Embedded Systems & Applications, pp. 315–320 (2011)
22. Benini, A., Mancini, A., Frontoni, E., Zingaretti, P., Longhi, S.: A simulation framework for coalition formation of unmanned aerial vehicles. In: 2011 19th Mediterranean Conference on Control Automation (MED), pp. 406–411 (2011)
23. SimplySim: SimplyCube. http://www.simplysim.net/ Accessed March 2011
24. De Agostino, M., Manzino, A.M., Piras, M.: Performances comparison of different mems-based imus. In: Position Location and Navigation Symposium (PLANS), 2010 IEEE/ION, pp. 187–201 (2010)
25. De Bento, M., Eissfeller, B., Machado, F.: How to deal with low performance imus in an integrated navigation system: Step by step. In: 2010 5th ESA Workshop on Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing (NAVITEC), pp. 1–10 (2010)
26. Nebot, E., Durrant-Whyte, H.: Initial calibration and alignment of low-cost inertial navigation units for land vehicle applications. J. Robot. Syst. **16**(2), 81–92 (1999)
27. Scaramuzza, D., Fraundorfer, F.: Visual odometry [tutorial]. IEEE Robot. Automat. Mag. **18**(4), 80–92 (2011)
28. Ascani, A., Frontoni, E., Mancini, A., Zingaretti, P.: Feature group matching for appearance-based localization. In: IROS, pp. 3933–3938 (2008)
29. Zhang, Z.: A flexible new technique for camera calibration. IEEE Trans. Pattern anal. Mach. Intell. **22**, 1330–1334 (2000)