Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

Actas de las IX Jomadas Argentinas de Robótica 15-17 de noviembre, Córdoba, Argentina

Validation of an IMU-camera fusion algorithm using an industrial robot

Validación de un algoritmo de fusión de IMU y cámara usando un robot industrial

Presentación: 25/09/2017

Aprobación: 02/12/2017

Gonzalo Perez-Paina
Centro de Investigación en Informática para la Ingeniería, Facultad Regional Córdoba, Universidad Tecnológica Nacional - Córdoba, Argentina
gperez@frc.utn.edu.ar

Claudio Paz
Centro de Investigación en Informática para la Ingeniería, Facultad Regional Córdoba, Universidad Tecnológica Nacional - Córdoba, Argentina
cpaz@frc.utn.edu.ar

Martín Pucheta
Centro de Investigación en Informática para la Ingeniería, Facultad Regional Córdoba, Universidad Tecnológica Nacional y Consejo Nacional de Investigaciones Científicas y Técnicas - Córdoba, Argentina
mpucheta@frc.utn.edu.ar

Bruno Bianchini
Centro de Investigación en Informática para la Ingeniería, Facultad Regional Córdoba, Universidad Tecnológica Nacional - Córdoba, Argentina
bruno.bian.26@gmail.com

Fernando Martínez
Centro de Investigación en Informática para la Ingeniería, Facultad Regional Córdoba, Universidad Tecnológica Nacional - Córdoba, Argentina
fermartinez51@gmail.com

Martín Nievas
Centro de Investigación en Informática para la Ingeniería, Facultad Regional Córdoba, Universidad Tecnológica Nacional - Córdoba, Argentina
mnievas@frc.utn.edu.ar

Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

Abstract

The integration of down-looking camera with an in-ertial measurement unit (IMU) sensor makes possible to provide a lightweight and low-cost pose estimation system for unmanned aerial vehicles (UAVs) and micro-UAVs (MAVs). Recently, the authors developed an algorithm for IMU and exteroceptive sensor fusion filter for position and orientation estimation. The aim of the estimation is to be used in the outer control loop of an UAV for position control. This work presents an experimental set up to test that algorithm using an industrial robot to produce accurate planar trajectories as a safe alternative to testing the algorithm on real UAVs. The results of the IMU-camera fusion estimation for linear positions and linear velocities show an error admissible to be integrated on real UAVs.

Keywords: sensor fusion, inertial measurement unit, monocular camera, industrial robot, error-state Kalman filter.

Resumen

La integración de una cámara orientada hacia abajo con un sensor de unidad de medición inercial (IMU, por las siglas en inglés de Inertial Measurement Unit) hace posible proporcionar un sistema de estimación de pose liviano y de bajo costo para vehículos aéreos no tripulados (UAVs, por Unmanned Aerial Vehicle) y micro-UAVs (MAVs, por Micro Unmanned Aerial Vehicle). Recientemente, los autores desarrollaron un algoritmo de un filtro de fusión de señales de una IMU y un sensor exteroceptivo para la estimación de la posición y la orientación. El objetivo de la estimación es utilizarlo en el bucle de control exterior de un UAV para el control de posición. Este trabajo presenta una configuración experimental para probar ese algoritmo utilizando un robot industrial para producir trayectorias planas precisas como una alternativa segura a probar el algoritmo en UAV reales. Los resultados de la estimación de fusión IMU-cámara para posiciones lineales y velocidades lineales muestran un error admisible para ser integrado en UAVs reales.

Palabras claves: fusión sensorial, unidad de medición inercial, cámara monocular, robot industrial, filtro de error de estado de Kalman.

I Introduction

The autonomous navigation of unmanned aerial vehicles (UAVs) in GPS-denied environments is currently possible by using controllers that employs an on-board computation of the feedback signals of the vehicle pose (position and orientation) using multiple sensors –e.g., IMUs (Inertial Measurement Unit), cameras (monocular, stereo, and RGB-D), and/or laser range finders– combined with efficient algorithms for data fusion that together minimize the uncertainty of the individual sensor measurements.

Concerning the state estimation for UAVs and MAVs, Kumar and Michael (2012) highlighted that the integration of camera sensors with IMUs enable a lightweight alternative to designs based on laser range finders or RGB-D cameras (that provide dense and rich three-dimensional information but have a trade-off with payload constraints and computational resources required). With the aim to (i) use low-weight and low-cost sensors, (ii) use efficient real-time algorithms to reduce computational resources, (iii) use on-board computation to avoid the low-latency and possible interruptions of the communication for off-board computation, and (iv) reduce the energy consumption to increase the autonomy, the authors developed a sensor fusion algorithm (PerezPaina et al., 2017) for IMU and camera sensors that is further experimentally evaluated in this work.

In order to intensively test a sensor fusion algorithm, the first option is to use a real UAV but involves a risk of damage if the algorithm fails while the flight takes place. The main idea of the present work is to test the algorithm on-line but in a detached way from the real UAV's flights. So it is necessary to generate an accurate reference motion using another facility, in this case, an industrial manipulator is chosen; see Fig.1.

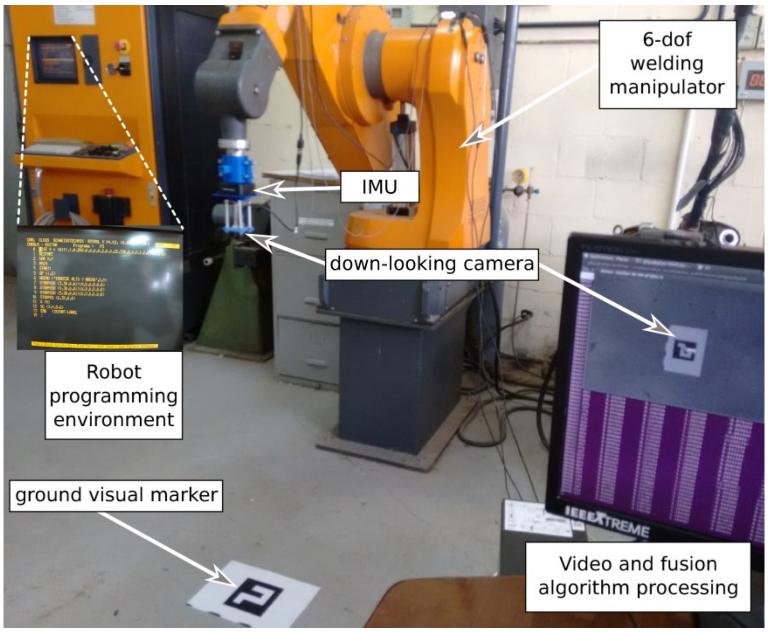

6-dof welding manipulator
IMU
down-looking camera
Robot programming environment
ground visual marker
Video and fusion algorithm processing

Fig. 1: Experimental setup to test the IMU-camera fusion estimation algorithm.

In this work, the proposed experimental set up is to generate motions for an IMU and a down-looking camera mounted on the end-effector of a robot manipulator to emulate the signals emitted by an UAV in motion. The measurements of the IMU and the down-looking camera are fused using the algorithm proposed PerezPaina et al. (2017). The aim of this work is to validate the sensor fusion for a motion parallel to the ground plane, including linear position and linear velocity, to be used in an UAV on-board controller, as the proposed Pucheta et al. (2016). The full pose estimation will be tackled in future work. The experiments show that the errors in the measurements are admissible to be integrated in the real system.

The article is organized as follows. Section II introduces the description of the robot manipulator used to generate the reference motion. Then, it describes the selection of the fiducial visual marker used to estimate the pose of the down-looking camera. Next, the rigid transformations used in the IMU-camera sensors are detailed. Section III briefly reviews the fusion estimation algorithm. Section IV describe the implementation and the obtained results for two different paths, and section V presents the conclusions and future work.

II Materials and methods

A. Welding manipulator

An industrial robot manipulator, CLOOS Romat 310, is used in this work to generate accurate trajectories. It is available from the Mechanical Engineering's Department of the FRC-UTN. This robot is a six- degrees-of-freedom serial anthropomorphic arm and is equipped with the ROTROL control system and a proprietary programming language named CAROLA. With respect to the average size of the welding robots existing in the market, its workspace is comparatively large (it is often combined with an additional translational axis to weld large mechanical parts). Its workspace is large enough to include reference trajectories for UAV calibration purposes.

The robot can be programmed by teaching some way-points to approximate the end-effector near the working volume, and from that point, several trajectories can be defined using geometric templates referenced to a predefined coordinate system. For instance, a program can be referenced to the home position, to a used-defined frame of reference, or can be relative to a frame located at the base of the robot; in this work the latter frame is chosen. The end-effector pose is defined by a 6-tuple of coordinates, 3 for the Cartesian positioning (x, y, z) with respect to a frame located at the base of the robot and 3 for the orientation (α, β, γ) relative to the home frame orientation. Because its welding capabilities, the programming language allows the user to define many characteristics for the interpolation of motions, several ways to approach the sharp corners of the trajectories in a controlled form, avoiding overlapping or oscillation through the target trajectories. This makes the robot an valuable research platform to generate accurate free-form paths in $SE(3)$.

Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

B. Visual marker for pose estimation

A fiducial visual marker on the ground is used to obtain an estimation of the absolute pose of the down-looking camera. To detect and track this marker is easier than to employ natural features. Visual marker with geometric forms like square are better than other shapes because they have four remarkable points (corners) that can be used to determine the pose with respect to the camera, while inner region can be used for identification. Different methods are used for identification and binary patterns are the most usual. Among the predefined pattern for visual markers available in the literature, the ArUco dictionary of markers (Garrido-Jurado et al., 2014), which maximize intermarker distance is chosen. This allow to reduce the size of the dictionary to the minimum needed, increasing the speed of detection, when more than one marker is used.

The ArUco algorithm is integrated with the OpenCV library (Bradski & Kaehler, 2008) to compute the rigid transformations between the camera and the maker frame systems.

C. Reference frame systems for the used setup

The reference systems of the used setup can be seen in Fig. 2, where $\{i\}$ is the inertial or navigation frame attached to the ground visual marker, $\{b\}$ the body frame (coincident with the IMU frame), and $\{c\}$ the camera frame.

The translation vector $\boldsymbol{t}_{ci}^{i} \in \mathbb{R}^3$ and the orientation matrix $\boldsymbol{R}_c^i \in SO3$ relating the navigation and the camera frames, are estimated by the visual algorithm. The translation vector $\boldsymbol{t}_{ib}^{i} \in \mathbb{R}^3$ and the orientation matrix $\boldsymbol{R}_b^i \in SO3$ relating the navigation and the body frame, are estimated by the fusion filter. Last, the relative pose given by the vector$\boldsymbol{t}_{cb}^{c} \in \mathbb{R}^3$ and the matrix $\boldsymbol{R}_b^c \in SO3$, relating the camera and body frames, is a known data measured by hand. As can be seen in Fig. 2, this relative pose is given only by a translation in the z coordinate and therefore $\boldsymbol{R}_b^c = \boldsymbol{I}$.

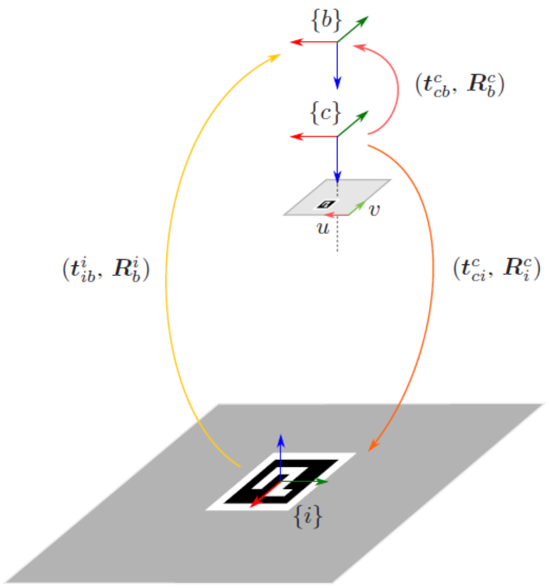

$\{b\}$
$\{c\}$
$(\boldsymbol{t}_{cb}^c, \boldsymbol{R}_b^c)$
$(\boldsymbol{t}_{ib}^i, \boldsymbol{R}_b^i)$
$(\boldsymbol{t}_{ci}^c, \boldsymbol{R}_i^c)$
u
v
$\{i\}$

Figure 2: Three frame systems and their rigid transformations represented by curved arrows: $\{c\}$ is the frame system located at the optical center of the camera, $\{i\}$ represents the inertial or navigation frame system, and $\{b\}$ is the frame with unknown pose to be referred with respect to $\{i\}$. For each frame, local arrows in red, green, and blue color represent the x, y, and z axes, respectively.

D. Reference frame transformation

Both the relative position and orientation between the camera reference system $\{c\}$, and the reference system of the ground marker attached to inertial frame $\{i\}$, can be written as $\boldsymbol{t}_{ci}^{c} \in \mathbb{R}^3$ and $\boldsymbol{R}_i^c \in SO3$, respectively (Ma et al., 2010). For vector $\boldsymbol{t}_{ci}^{c}$, the sub-index ci indicates the starting $\{c\}$, and target $\{i\}$, reference systems. The supra-index indicates the frame in which the coordinates of this vector are expressed. Concerning the convention for

Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

the orientation, the matrix $\boldsymbol{R}_i^c$, uniquely represents the orientation of the reference system $\{i\}$ with respect to the camera reference system $\{c\}$. This matrix can also be used to change the coordinates of a vector from the $\{i\}$ reference system to the camera reference system $\{c\}$ (Pucheta et al., 2014).

As was stated, using the visual algorithm the pose given by $\boldsymbol{t}_{ci}^c$ and $\boldsymbol{R}_i^c$ is obtained, for which the inverted transformation needed by the fusion filter $\boldsymbol{t}_{ic}^i$ and $\boldsymbol{R}_c^i$ is computed.

In order to compute the elements of the unknown pose, $\boldsymbol{R}_b^i$ and $\boldsymbol{t}_{ib}^i$, a geometric analysis is performed. Vector addition of relative positions can be formulated as

$$\boldsymbol{t}_{ib}^i = \boldsymbol{t}_{ic}^i + \boldsymbol{t}_{cb}^i \tag{1}$$

where, $\boldsymbol{t}_{ic}^i$ and $\boldsymbol{t}_{cb}^i$ respectively represents the same vectors as $\boldsymbol{t}_{ci}^c$ (although in different sense) and $\boldsymbol{t}_{cb}^c$ but referred to the coordinate system $\{i\}$ to make the sum consistent. This pair of positions can be obtained by using the change of coordinates matrix $\boldsymbol{R}_c^i$, which is the inverse (and also the transpose for being orthogonal) of $\boldsymbol{R}_i^c$, computed by the detection algorithm. Then, (1) can be written as

$$\boldsymbol{t}_{ib}^i = \boldsymbol{R}_c^i \boldsymbol{t}_{ic}^c + \boldsymbol{R}_c^i \boldsymbol{t}_{cb}^c \tag{2}$$

Provided that

$$\boldsymbol{t}_{ic}^c = -\boldsymbol{t}_{ci}^c, \tag{3}$$

(1) can also be written as

$$\boldsymbol{t}_{ib}^i = (\boldsymbol{R}_i^c)^T (\boldsymbol{t}_{cb}^c - \boldsymbol{t}_{ci}^c) \tag{4}$$

On the other hand, Fig. 2 also depicts that a vector represented in the frame system $\{b\}$ can be changed of basis to be referred to the system $\{i\}$, and that is equivalent to a indirect change from $\{b\}$ to $\{c\}$ followed by a change from $\{c\}$ to $\{i\}$.

An arbitrary vector $\boldsymbol{v}^b$ can be changed of reference system from $\{b\}$ to $\{i\}$ by means of

$$\boldsymbol{v}^i = \boldsymbol{R}_b^i \boldsymbol{v}^b \tag{5}$$

or, using an intermediate frame, by computing

$$\boldsymbol{v}^i = \boldsymbol{R}_c^i \boldsymbol{R}_b^c \boldsymbol{v}^b \tag{6}$$

where, clearly, the matrix that represents the orientation of the frame system $\{b\}$ with respect to the frame system $\{i\}$ is the same that produces the change of basis for a vector from $\{b\}$ to $\{i\}$. Therefore, the orientation of frame $\{b\}$ with respect to frame $\{i\}$ can be written as

$$\boldsymbol{R}_b^i = \boldsymbol{R}_c^i \boldsymbol{R}_b^c \tag{7}$$

or, using the data supplied by the algorithm described by Garrido-Jurado et al. (2014), as

$$\boldsymbol{R}_b^i = (\boldsymbol{R}_i^c)^T \boldsymbol{R}_b^c. \tag{8}$$

Figure 3 shows real setup, similar to the schematic view in Fig.2, indicating the three reference system evolved, $\{i\}$, $\{c\}$, and $\{b\}$. The 3D printed sensor holder, which rigidly attach the camera and IMU (fixed to the manipulator end-effector) can be seen in the zoom area.

Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

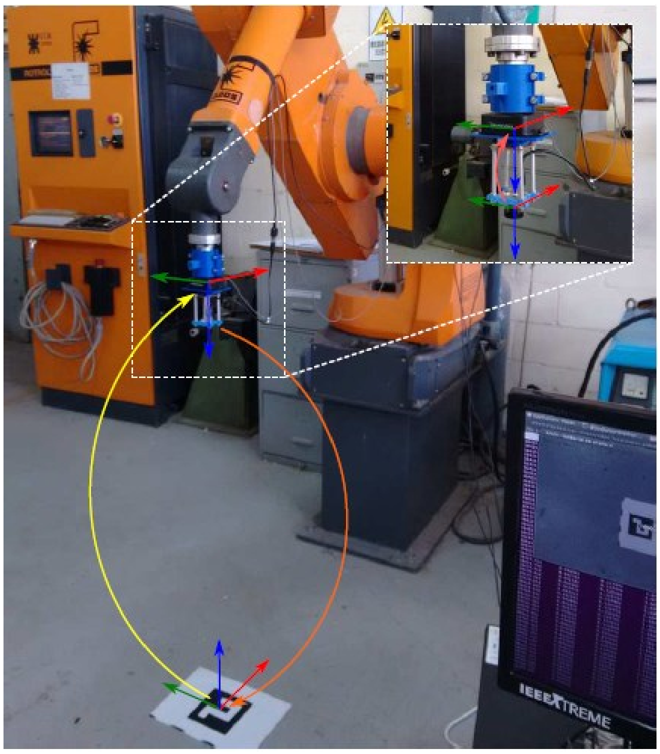


Figure 3: Yellow and orange arrows represent relative position and orientation between different systems. Red, green, and blue arrows represent the *x*, *y*, and *z* axis, respectively.

III. IMU-Camera fusion

The implemented sensor fusion algorithm is based on the Error-State Kalman Filter (ESKF), which performs a loosely coupled sensor fusion (Madyastha et al., 2011; Solà, 2015). The inertial unit is composed of a three axis accelerometer and a three axis gyroscope. The implemented filter is based on the described by Perez Paina et al. (2017).

A. Error-state Kalman Filter

In the ESKF, the state is expressed as a nominal value plus an error or perturbation, $\boldsymbol{x} = \boldsymbol{x}_n \oplus \delta\boldsymbol{x}$, where the symbol $\oplus$ (composition) is an ordinary addition for position and velocity, and a quaternion product for the orientation. The ESKF performs the nominal state propagation while estimating the error state used to correct this integration.

The formulation of the ESKF is based on the system stochastic model, given by

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) \tag{9}$$
$$\boldsymbol{z} = h(\boldsymbol{x}, \boldsymbol{v}), \tag{10}$$

where the state has to be expressed in terms of the nominal and the error parts.

The nominal state $\boldsymbol{x} = [\boldsymbol{p}^T \quad \boldsymbol{v}^T \quad \boldsymbol{q}^T]^T$ is composed of the position $\boldsymbol{p}$, the linear velocity $\boldsymbol{v}$, and the orientation expressed as a unit quaternion $\boldsymbol{q}$, all with respect to an inertial frame. The inertial measurements are used as input in the process model (9), with $\boldsymbol{u} = [\boldsymbol{a}_m^T \quad \boldsymbol{\omega}_m^T]^T$. On the other hand, the exteroceptive sensor measurement of (10), given here by a visual monocular pose estimation, is composed of the absolute position and orientation, i.e. $\boldsymbol{z}_i = [\boldsymbol{p}_i^T \quad \boldsymbol{q}_i^T]^T$.

The prediction stage of the ESKF filter is given by

$$\hat{\boldsymbol{x}}_k^- = f(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_{k-1}, \boldsymbol{0}) \tag{11}$$
$$\boldsymbol{P}_k^- = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}\boldsymbol{F}_{k-1}^T + \boldsymbol{Q}_{k-1} \tag{12}$$

where (11) is obtained from the discretization of (9) and is used to integrate the nominal state from the inertial measurements. Equation (12) propagates the covariance matrix representing the estimation uncertainty and incorporates the sensor noise of (9), $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})$, which has also to be obtained from the continuous-time model.

Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

On the other hand, the update stage of the ESKF is given by

$$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k^T (\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{C}_k)^{-1} \tag{13}$$

$$\widehat{\boldsymbol{x}} = \widehat{\boldsymbol{x}}_k^- \oplus \boldsymbol{K}_k (z_k \ominus h(\widehat{\boldsymbol{x}}_k^-)) \tag{14}$$

$$\boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{P}_k^- \tag{15}$$

which uses the camera pose measurement z in (14), and the sensor model given by (10). Equation (13) incorporates the measurement noise of (10), $\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$.

As usual, the filter orientation error is denoted in its minimal representation, avoiding redundant parameters that can produce singularities when applying the needed additional constraints. In the context of the Kalman filter it is also known as a multiplicative error model (Markley, 2004; Markley, 2003). Thus, the filter estimation error is

$$\boldsymbol{x} \ominus \widehat{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{p} - \widehat{\boldsymbol{p}} \\ \boldsymbol{v} - \widehat{\boldsymbol{v}} \\ \widehat{\boldsymbol{q}}^* \otimes \boldsymbol{q} \end{bmatrix} = \begin{bmatrix} \delta\boldsymbol{p} \\ \delta\boldsymbol{v} \\ \delta\boldsymbol{q} \end{bmatrix} \equiv \delta\boldsymbol{x}, \tag{16}$$

where for a small orientation error, $\delta\boldsymbol{q}$ can be expressed as

$$\delta\boldsymbol{q} = \begin{bmatrix} \delta\boldsymbol{q}_w \\ \delta\boldsymbol{q}_v \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{1}{2}\delta\boldsymbol{\theta} \end{bmatrix}, \tag{17}$$

with $\delta\boldsymbol{\theta}$ being the minimum orientation error representation. Then, the error $\delta\boldsymbol{x}$ is used to compute the covariance matrix $\boldsymbol{P} = \mathbb{E}[\delta\boldsymbol{x} \quad \delta\boldsymbol{x}^T]$, which describes the uncertainty in the filter estimation required in (12), (13), and (15).

B. Process model based on inertial sensor measurements

The kinematic model used for integrating the inertial measurements is applied in the prediction stage of the estimation filter. For the Kalman filter implementation, the discrete-time kinematic model is derived from the continuous-time model, which for the case of the error state Kalman filter has to be separated in: (1) model for the nominal state integration, and (2) model for the uncertainty propagation.

1) Continuous-time model: The inertial sensor based kinematic continuous-time model is given by

$$\dot{\boldsymbol{p}} = \boldsymbol{v}, \tag{18}$$

$$\dot{\boldsymbol{v}} = \boldsymbol{R}(\boldsymbol{q})\boldsymbol{a} + \boldsymbol{g}_n, \tag{19}$$

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}, \tag{20}$$

where $\boldsymbol{R}(\boldsymbol{q})$ is the matrix representation of the orientation given by the unit quaternion $\boldsymbol{q}$, $\boldsymbol{g}_n$ is the gravity acceleration expressed in the inertial frame, the symbol $\otimes$ represents the Hamilton product (Trawny and Roumeliotis, 2005) where the first quaternion component is the scalar value.

The accelerometer and gyroscope inertial sensors model are

$$\boldsymbol{a}_m = \boldsymbol{a} + \boldsymbol{w}_a \tag{21}$$

$$\boldsymbol{\omega}_m = \boldsymbol{\omega} + \boldsymbol{w}_\omega \tag{22}$$

where the subscript m stands for measured value, which are composed of the true value plus an additive Gaussian noise $\boldsymbol{w}_a \sim \mathcal{N}(\boldsymbol{0}, \sigma_a^2\boldsymbol{I})$ and $\boldsymbol{w}_\omega \sim \mathcal{N}(\boldsymbol{0}, \sigma_\omega^2\boldsymbol{I})$.

By replacing (21) in (19) and (22) in (20), the continuous-time stochastic model of (9) is obtained, which gives

$$\dot{\boldsymbol{p}} = \boldsymbol{v}, \tag{23}$$

$$\dot{\boldsymbol{v}} = \boldsymbol{R}(\boldsymbol{q})(\boldsymbol{a}_m - \boldsymbol{w}_a) + \boldsymbol{g}_n, \tag{24}$$

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ (\boldsymbol{\omega}_m - \boldsymbol{w}_\omega) \end{bmatrix}, \tag{25}$$

Then, the nominal state evolution is given by

$$\dot{\boldsymbol{p}}_n = \boldsymbol{v}_n, \tag{26}$$

$$\dot{\boldsymbol{v}}_n = \boldsymbol{R}(\boldsymbol{q}_n)\boldsymbol{a}_n + \boldsymbol{g}_n, \tag{27}$$

$$\dot{\boldsymbol{q}}_n = \frac{1}{2}\boldsymbol{q}_n \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega}_n \end{bmatrix}, \tag{28}$$

and the error state by

$$\delta\dot{\boldsymbol{p}} = \boldsymbol{v}, \tag{29}$$
$$\delta\dot{\boldsymbol{v}} = -\boldsymbol{R}(\boldsymbol{q}_n)\lfloor \boldsymbol{a}_{m\times}\rfloor\delta\boldsymbol{\theta} + \boldsymbol{R}(\boldsymbol{q}_n)\boldsymbol{w}_a, \tag{30}$$
$$\delta\dot{\boldsymbol{\theta}} = -\lfloor \boldsymbol{\omega}_{m\times}\rfloor\delta\boldsymbol{\theta} - \boldsymbol{w}_\omega, \tag{31}$$

The derivation of (26) to (31) follows Solà (2015) and Trawny & Roumeliotis (2005).

As can be seen, the error state model of (29) to (31) is linear, which can be expressed as $\delta\dot{\boldsymbol{\theta}}$

$$\delta\dot{\boldsymbol{x}} = \boldsymbol{A}\delta\boldsymbol{x} + \boldsymbol{D}\boldsymbol{w} \tag{32}$$

with $\boldsymbol{w} = [\boldsymbol{w}_a^T \quad \boldsymbol{w}_\omega^T]^T$, and

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & -\boldsymbol{R}\lfloor \boldsymbol{a}_{m\times}\rfloor \\ \boldsymbol{0} & \boldsymbol{0} & -\lfloor \boldsymbol{\omega}_{m\times}\rfloor \end{bmatrix}, \boldsymbol{D} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ -\boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{I} \end{bmatrix}, \tag{33}$$

with $\boldsymbol{R} \equiv \boldsymbol{R}(\boldsymbol{q})$.

2) Discrete-time model: In order to obtain the discrete-time model, here it is proposed to use the Euler integral solution of (26)-(31) for the nominal state and uncertainty propagation, respectively. The discrete-time evolution of the nominal state (omitting the subscript *n*), is then given by

$$\boldsymbol{p}_k = \boldsymbol{p}_{k-1}\Delta t\, \boldsymbol{v}_{k-1}, \tag{34}$$
$$\boldsymbol{v}_k = \boldsymbol{v}_{k-1} + \Delta t \boldsymbol{R}(\boldsymbol{q}_{k-1})\boldsymbol{a}_{m,k-1} + \Delta t \boldsymbol{g}_n, \tag{35}$$
$$\boldsymbol{q}_k = \left(\boldsymbol{I} + \frac{\Delta t}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_{m,k-1})\right)\boldsymbol{q}_{k-1}, \tag{36}$$

where $\Delta t = t_k - t_{k-1}$ and

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} \boldsymbol{0} & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & \lfloor \boldsymbol{\omega}_\times \rfloor \end{bmatrix}$$

is a 4× 4 skew-symmetric matrix. Therefore, the state transition matrix of (12) results

$$\boldsymbol{F}_k = e^{\boldsymbol{A}_k(t_k - t_{k-1})} = \boldsymbol{I} + \Delta t \boldsymbol{A}_k, \tag{37}$$

where $\boldsymbol{A}$ is given by (33).

The discrete-time covariance matrix of the process noise is given by Simon (2006)

$$\boldsymbol{Q}_{k-1} = \int_{t_{k-1}}^{t_k} e^{\boldsymbol{A}(t_k-\tau)}\boldsymbol{D}\boldsymbol{Q}^c\boldsymbol{D}^T e^{\boldsymbol{A}(t_k-\tau)}d\tau. \tag{38}$$

A detailed representation of the sub-matrices of (38) can be found in Perez Paina et al. (2016).

C. Observation model based on camera measurements

The measurement vector of the visual pose estimation algorithm is composed of the position and orientation with respect to the inertial frame, i.e. $\boldsymbol{z}_i = [\boldsymbol{p}_i^T \quad \boldsymbol{q}_i^T]^T$. Hence, the measurement function is $\boldsymbol{z}_i = h(\boldsymbol{x}) + \boldsymbol{v}$, where $\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$ is the additive Gaussian noise. In order to compute the filter innovation $\boldsymbol{\nu}$, it is necessary to express the orientation using an unit quaternion, such that $\boldsymbol{\nu} = \boldsymbol{z} \ominus \hat{\boldsymbol{z}}^-$ can be computed as an ordinary subtraction for the position and using the quaternion product for the orientation. Given the nominal predicted state, the measurement prediction is $\hat{\boldsymbol{z}}^- = h(\hat{\boldsymbol{x}}_n^-)$, and the innovation can be computed as

$$\begin{aligned} \boldsymbol{\nu}_i = \boldsymbol{z}_i \ominus \hat{\boldsymbol{z}}_i^- &= \begin{bmatrix} \boldsymbol{p}_i \\ \mathbf{q}_i \end{bmatrix} \ominus \begin{bmatrix} \hat{\boldsymbol{p}}_i^- \\ \hat{\boldsymbol{q}}_i^- \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{p}_i - \hat{\boldsymbol{p}}_i^- \\ (\hat{\boldsymbol{q}}_i^-)^* \otimes \mathbf{q}_i \end{bmatrix} = \begin{bmatrix} \delta\boldsymbol{p}_i \\ \delta\mathbf{q}_i \end{bmatrix}, \end{aligned} \tag{39}$$

and considering a small orientation error

$$\delta\boldsymbol{q}_i = \begin{bmatrix} \delta\boldsymbol{q}_{i,w} \\ \delta\boldsymbol{q}_{i,v} \end{bmatrix} \approx \begin{bmatrix} 1 \\ \frac{1}{2}\delta\boldsymbol{\theta}_i \end{bmatrix}, \tag{40}$$

Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

then, the minimal representation of the innovation is $\delta \boldsymbol{v}_i = [\delta \boldsymbol{p}_i^T \quad \delta \boldsymbol{\theta}_i^T]^T$.

On the other hand, the Jacobian matrix $\boldsymbol{H}$ of the measurement model, used in (13) and (15), has to be defined. Given the visual algorithm providing absolute position and orientation measurements, this matrix results

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}, \tag{41}$$

The covariance of the camera measurements, used in (13), is $\boldsymbol{C} = \text{diag}(\boldsymbol{C}_p, \boldsymbol{C}_q)$, which is composed of the covariance matrix to describe the uncertainty $\boldsymbol{C}_p = \sigma_{p_i}^2 \boldsymbol{I}$ in the position and $\boldsymbol{C}_{\gamma_i} = \sigma_{\gamma_i}^2 \boldsymbol{I}$ for the orientation expressed with Euler angles. For a more complex sensor, the measurement Jacobian matrix can be obtained as explained in Solà (2015).

IV. Implementation and Results

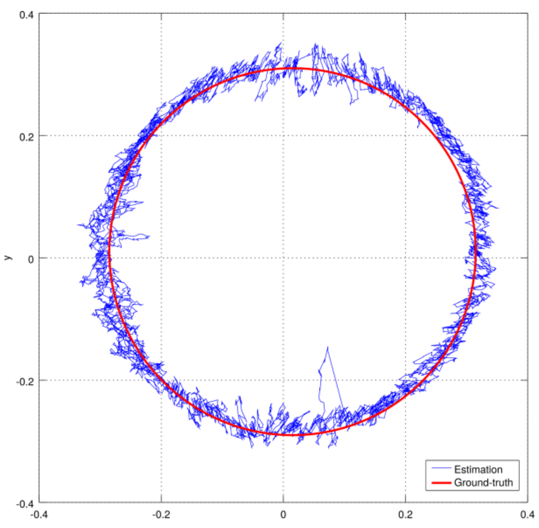
A. Implementation

The experimental results were obtained using the following hardware equipment: inertial measurement unit from Microstrain model 3DM-GX1 where raw data was used, camera from The Imaging Source model DMM22BUC03 (USB 2.0, 640x480 px), with optical lens also from The Imaging Source model TBN3.5C-3MP (low distortion board lens M12x0.5mm, focal length 3.5mm). The fusion filter was implemented in C++ programming language using the computer vision library OpenCV version 3.3.0, with the contribution module (also v3.3.0) for the ArUco algorithm implementation, and the Eigen library v3.3 for the filter matrix operations.

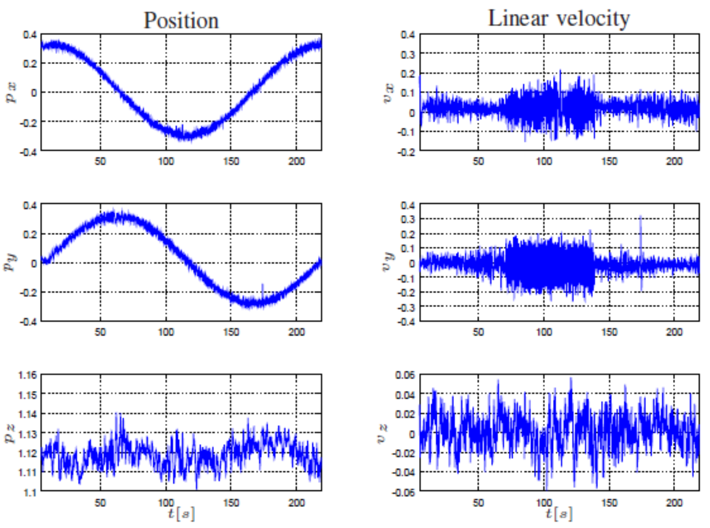
B. Results

The experimental evaluation of the fusion filter was carried out using two different paths at constant linear velocity: a circle, and a square one. Figure 4 shows the estimation result with the circular path. In Fig. 4a the xy path can be observed, the red line represents the real path generated by the robot manipulator, and the blue one is the estimation given by the fusion filter, whereas Fig. 4b shows the position in meters and the linear velocity in meter per seconds against the time given in seconds. Figure 5 shows the same as Fig. 4 but for the square path.

As it can appreciated in the top view of the xy-path given in Fig. 4a and 5a, the estimation presents more noise near the $x = 0$ and $y = 0$ planes. When approaching these particular points, the used visual algorithm gives an orientation estimation which oscillates between both sides of x or y axes. Then, when the camera measurement is inverted to obtain the pose with respect to the navigation frame, the orientation oscillation affects more the error in the translation estimation. The elimination of these effects will be the focus of next research.



(a)



(b)

Fig. 4: Circular path: (a) Shows the reference path in red color and the estimated one in blue, and (b) shows the position and linear velocity vs. time.

Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

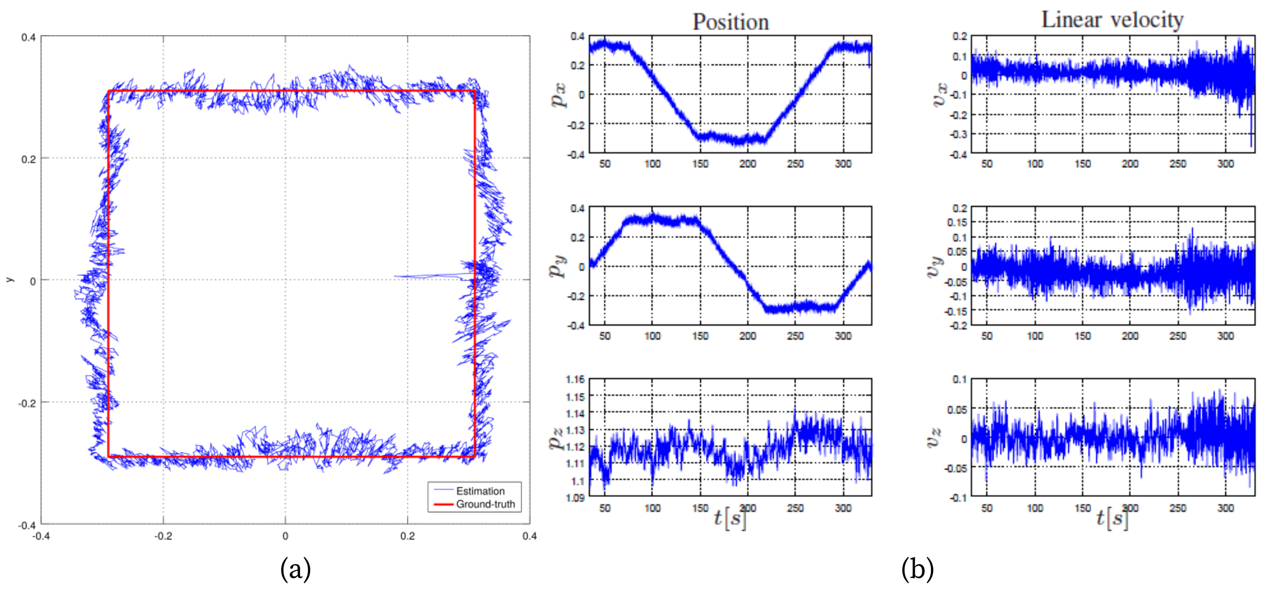


Fig. 5: Square path: (a) Shows the reference path in red color and the estimated one in blue, and (b) shows the position and linear velocity vs. time.

Conclusions

This work presented preliminary results of experiments achieved to test a fusion algorithm to integrate the measurements of a down-looking camera with an inertial measurement unit sensor, which will be take part of the full pose feedback signal of a UAV control system. An industrial robot was used to generate planar trajectories at constant linear velocity, a circle and an square, to test the algorithm for position estimation. The results for each task show position and velocity estimation errors admissible to be integrated on real UAVs. Next experiments will include the orientation estimation to test the full pose estimation. It is also expected that the level of noise in the real UAV will be higher than the produced by the robot.

Acknowledgments

This work was supported by the Universidad Tecnológica Nacional (Projects PID-UTN 3935, UTI3832TC, AMUTICO0007819TC and CCUTNCO0007805). The authors acknowledge to Nelson Vallejos, from the Mechanical Engineering Department, UTN-FRC, for assisting us with the robot programming issues.

Revista Tecnología y Ciencia
DOI: https://doi.org/10.33414/rtyc.37.101-111.2020 - ISSN 1666-6933

Referencias

Bradski, G. & Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. Boston, MA, USA: O'Reilly Media, Incorporated.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., & Marín-Jiménez, M. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. *PatternRecognition*, **47**(6):2280–2292. doi: 10.1016/j.patcog.2014.01.005

Kumar, V. & Michael, N. (2012). Opportunities and challenges with autonomous micro aerial vehicles. *The International Journal of Robotics Research*, **31**(11):1279–1291. doi: 10.1177/0278364912455954

Ma, Y., Soatto, S., Kosecká, J., & Sastry, S. S. (2010). *An Invitation to 3-D Vision: From Images to Geometric Models*. Interdisciplinary Applied Mathematics, Berlin: Springer.

Madyastha, V., Ravindra, V., Mallikarjunan, S., & Goyal, A. (2011). Extended Kalman Filter vs. Error State Kalman Filter for Aircraft Attitude Estimation. *Guidance, Navigation, and Control and Co-located Conferences*. 8-11 August, Portland, Oregon. doi: 10.2514/6.2011-6615

Markley, F. L. (2003). Attitude Error Representations fo Kalman Filtering. *Journal of Guidance, Control, and Dynamics*, **26**(2):311–317. doi: 10.2514/2.5048

Markley, F. L. (2004). Multiplicative vs. additive filtering for spacecraft attitude determination. In: 6th Conference on *Dynamics and Control of Systems and Structures in Space (DCSSS)*. Riomaggiore, Italy: NACA.

Perez Paina, G., Paz, C., Kulich, M., Saska, M., & Araguás, G. (2016). Fusion of Monocular Visual-Inertial Measurements for Three Dimensional Pose Estimation, volume 9991 of *Lecture Notes in Computer Science*, chapter 20, pages 242–260. Cham, Switzerland: Springer International Publishing.

Perez Paina, G., Pucheta, M., & Paz, C. (2017). IMU- and exteroceptive sensor-based fusion for UAV control. In: Proceedings of the XVI Workshow on Information Processing and Control RPIC 2017, pages 1-6. Mar del Plata, Argentina: AADECA. doi: 10.23919/RPIC.2017.8214335

Pucheta, M. A., Alberto, N., Paz, C., & Perez Paina, G. (2016). Trajectory planning for an unmanned quadrotor. In: Giusti, S., Pucheta, M., and Storti, M. (eds.), *Mecánica Computacional Vol. XXXIV,* Proc. of the XXII Congreso sobre Métodos Numéricos y sus Aplicaciones ENIEF 2016, pages 2809–2824, Córdoba, Argentina: AMCA.

Pucheta, M. A., Paz, C. J., & Pereyra, M. E. (2014). Representaciones cinemáticas de orientación y ecuaciones de estimación. In: Bertolino, G., Cantero, M., Storti, M. and Teruel, F. (eds.) *Mecánica Computacional Vol. XXXIII*, Proc. of the XXI Congreso sobre Métodos Numéricos y sus Aplicaciones ENIEF 2014, pages 2303–2324, Bariloche, Argentina: AMCA.

Simon, D. (2006). *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. New York, USA: Wiley-Interscience.

Solà, J. (2015). Quaternion kinematics for the error-state Kalman filter. Preprint *arXiv*: 1711.02508.

Trawny, N. & Roumeliotis, S.I. (2005). Indirect Kalman filter for 3D attitude estimation. *Technical Report 2005-002 Rev. 57,* University of Minnesota, Department of Computer Science & Engineering.