Journal of Intelligent & Robotic Systems (2021) 102:18
https://doi.org/10.1007/s10846-021-01377-3

REGULAR PAPER

An Embedded Quaternion-Based Extended Kalman Filter Pose Estimation for Six Degrees of Freedom Systems

Rodrigo Alves Medeiros[1] · Guilherme Araujo Pimentel[1,2] · Rafael Garibotti[1]

Received: 31 March 2020 / Accepted: 26 March 2021

Abstract

This paper proposes a formulation of quaternion-based Extended Kalman Filter pose estimation for six degrees of freedom systems embedded in an FPGA with commercial processors. Our approach uses the fusion of a camera and an inertial measurement unit to estimate simultaneously the position and the orientation of the system of interest. In addition, a Stewart platform is used to validate and evaluate the estimated pose. Although this work considers the use of common low-cost sensors and the use of markers with simple geometry, the results show excellent performance of the developed filter, being able to estimate the pose and orientation with an error below 8.14 mm and 0.63º, respectively. Furthermore, the effectiveness of the approach has also been evaluated, showing that the filter is able to converge quickly when the markers are retrieved after a loss of camera data for a short period of time.

Keywords Extended Kalman filter · Stewart platform · Quaternion · Pose estimation · Embedded systems · FPGA

1 Introduction

In the last decades, we have seen the great development and adoption of automation by the industry. Today, systems with six degrees of freedom (6–DOF) have been widely used in several fields of industry, such as automotive [1–3], entertainment [4, 5] and military [6], due to the complexity of the tasks they can perform [7–9]. Because of its relevance, 6–DOF systems received special attention from academy researchers, mainly on the estimation of the system position and orientation, also known as pose estimation.

To estimate the pose of a 6–DOF system, a set of sensors capable of measuring the angular velocity and the acceleration of a rigid-body are commonly used, and in some cases also the direction of the magnetic field. The association of these sensors for system orientation purposes into a single unit is called the Inertial Measurement Unit (IMU). These devices are suitable to provide measurements at a frequency of up to 1 kHz. However, this may cause that these sensors could be corrupted by additive and offset noise, also known as measurement bias [10]. Furthermore, the signal usually suffer from a phenomenon known as *Drift*, caused by the accumulation of errors introduced in the measurements. To overcome this problem, other sensors are normally used in conjunction with the IMU. This fusion of sensors is intended to signal correction [11].

The most used methodology of estimation and fusion of sensors is the Extended Kalman Filter (EKF), due to its statistical properties, in addition to the versatility for different dynamic models and sensor set [12, 13]. This filter has two different steps, prediction and update. In the prediction step, a linearized model of the system is used to infer the desired output and, later, in the update step, a correction is performed through the sensors. In addition, most EKF works are based on Euler angles to represent the orientation dynamics [14]. However, as it uses trigonometric relations, its computational cost becomes quite high.

To improve the quality of the algorithm, we propose an EKF algorithm based on quaternions to estimate the pose of a 6–DOF system, considering the fusion of the signal coming from an IMU and a camera (i.e., from visual

✉ Rafael Garibotti
rafael.garibotti@pucrs.br

Rodrigo Alves Medeiros
rodrigo.medeiros@acad.pucrs.br

Guilherme Araujo Pimentel
guilherme.araujopimentel@umons.ac.be

[1] School of Technology – Pontifical Catholic University of Rio Grande do Sul, PUCRS, Av. Ipiranga, 6681, Porto Alegre, Brazil

[2] Systems, Estimation, Control and Optimization Department University of Mons, UMONS, 31 Boulevard Dolez, Mons, Belgium

Published online: 22 April 2021

marker data). Quaternions are present in different areas of mathematics, and has particular use for three-dimensional rotations, as they provide an alternative to traditional Euler angles. Under this perspective, the quaternions' advantage in relation to Euler angles is by changing trigonometric nonlinearities for polynomial nonlinearities, which reduces the computational effort involved and also avoids the *Gimbal Lock* problem [15, 16].

To demonstrate the applicability of our solution, we implement the EKF algorithm on a Stewart platform, which is the most known parallel mechanism used in many robotic fields with 6–DOF [17]. The Stewart platform consists of a fixed and a movable platform connected by six linear actuators [18]. Figure 1 shows some real-world applications where the Stewart platform has been used, such as flight simulators [18]; precision surgery [19]; interferometer [20]; offshore cargo transfer mechanism [21]; and manufacturing [22]. This diverse and growing environment in which 6–DOF systems are used highlights the need for increasingly efficient systems.

Another advantage of our proposed solution is that it has been embedded in an FPGA device with commercial processors, which improves performance, reduces power consumption and facilitates reuse in modern embedded architectures [23]. Furthermore, our implementation also stands out for improving the robustness and efficiency of 6–DOF systems, as our approach estimates simultaneously all system states – position and orientation on each of the three-dimensional axis.

The rest of this paper is organized as follows. Section 2 presents related works that propose algorithms to solve the pose estimation problem of 6–DOF systems. Next, Section 3 details the mathematical model used in this work. Following, Section 4 explains our approach to estimating the pose of the Stewart platform. Section 5 explores the promoted hardware 6–DOF pose estimation using Extended Kalman Filter. Finally, Section 6 points out the conclusions and future work.

2 Literature Review

The pose estimation in parallel mechanisms can be obtained by the difference between the position predicted by a kinematic model and the position measured by sensors [24]. The most common approach is to convert the legs length into the position and orientation of the end-effector, called the forward kinematics approach [25]. However, the main disadvantage when dealing with forward kinematics, in parallel mechanisms, is the difficulty of obtaining its analytical solution [26]. Some studies considering the geometric aspect of a Stewart platform (i.e., a 6–DOF parallel mechanism) have obtained high-degree polynomials with multiple solutions from one set of encoder measurements. For example, Cardona [17] shows that without considering any constraints, it is possible to find 4096 solutions for the same set of encoder measurements. However, taking some considerations, one can find in the literature techniques that obtain 43 [27] or even 40 possible solutions [28]. This concern is because the Stewart platform is an over-constrained system in which joint movements are not independent of each other, thus requiring a coordinated motion for all joints [29]. Thus, it is difficult to obtain a single solution due to the complex manipulation of the mathematical model equations [17].

To overcome this problem, some authors have presented numerical techniques to solve, through optimization, the forward kinematics to obtain the pose estimation of 6–DOF platforms. Song and Kwon [30] presented a tetrahedron-based formulation approach for real-time computation of the forward kinematics. Another one of these pose estimation numerical approaches is the super-twisting observer algorithm proposed by Kumar et al. [31]. This application uses the Lagrangian method to estimate a 6–DOF platform dynamics based on the measurements of its leg lengths. Another proposal is presented by Yang et al. [32], which proposed a Newton-Raphson-based

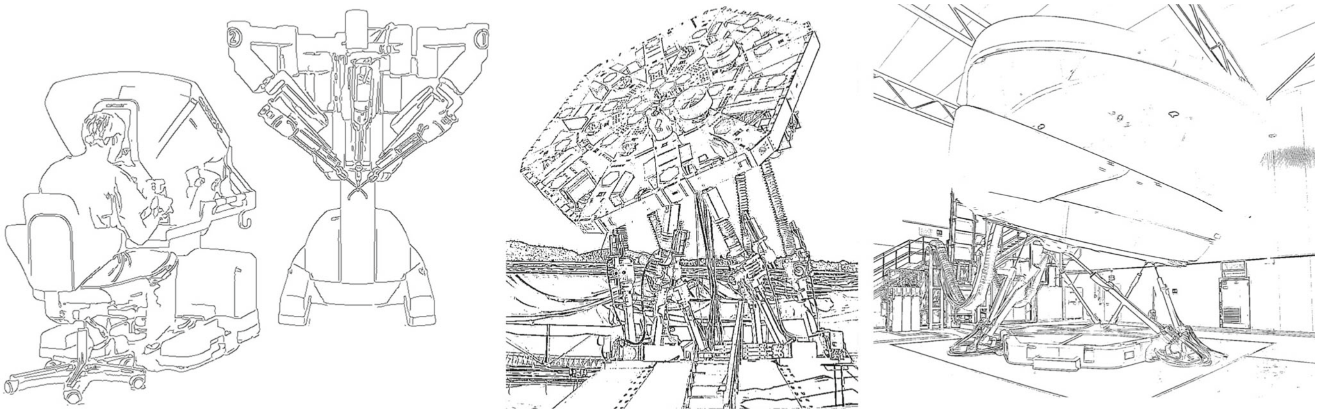


Fig. 1 Real-world Stewart platforms such as precision surgery (left), interferometer (center) and flight simulators (right)

numerical method for real-time pose estimation, called the Global Newton-Raphson with Monotonic Descent algorithm (GNRMD). This algorithm uses an orientation parameterization based on Euler angles with an analytical approach based on the geometric configuration of a Stewart platform to estimate the system pose. Nonetheless, this approach requires an initial estimate value, which in some cases ends in an infinite loop. Although the presented numerical approaches contribute to estimate the pose of a 6–DOF platform, the dependence on a single source of sensors is a weakness of these algorithms, which can produce large estimation errors.

In order to mitigate precision errors in solving the forward kinematic problem, more information on the end-effector pose is needed. On the one hand, a widely used procedure is the vision-based method. Rendón-Mancha et al. [33] and Coronado et al. [29] presented a camera-space manipulation method, which relies on the estimation of the relationship between the position of the visual markers in the robot's manipulator and their corresponding position in the images taken by at least two cameras and a linear model. In addition, Zuo et al. [34] used the stereo vision approach to measure the pose of the Stewart platform in real-time. Yoon et al. [35] proposed a Kalman-Filter-based visual tracking algorithm that provides 3D pose estimates of a rigid object using a single camera. The main inconvenience of these vision-based approaches is the need for very specific cameras and high processing power.

On the other hand, the vision-sensor based method is gaining momentum to solve these difficulties. Our approach follows this trend: we have an IMU with a gyroscope and an accelerometer; and a single camera that uses markers with different colors and simple geometry to minimize computational cost. To the best of our knowledge, very few works present similar sensing technology to obtain the pose estimation of 6–DOF systems. For example, Araguás et al. [36] present only the orientation estimation using a camera and inertial sensors for a hovering UAV; Erdem and Ercan [37] present topologies for the EKF implementation; while Nützi et al. [38] use the vision-sensor based method to estimate some variables instead of the pose estimation.

With the focus on the pose estimation, Mirzaei and Roumeliotis [39] presented an EKF-based algorithm to estimate the pose of vision-aided inertial navigation systems, using measurements from a single camera and an IMU. Similarly, Du and Zhang [40] proposed a method that incorporates a Kalman filter and a particle filter to automatically estimate the robot's poses during manipulation. However, both studies focus on camera calibration and neither is implemented on the Stewart platform, showing no experimental results to compare the ground truth with this work. More recently, Miletović et al. [41] presented a method to reconstruct the kinematic state of a Stewart platform based on the Unscented Kalman Filter (UKF). This work stands out in the literature, due to the formulation and validation of the method using a real Stewart platform. However, like other sensor-based works, reliance on a single data source can lead to a slightly larger error or a weakness in estimating the pose with data loss.

Unlike presented works, this paper proposes the formulation of a quaternion-based Extended Kalman Filter pose estimation with IMU and visual marker data for six degrees of freedom systems, with the aim to (*1*) accurately estimate the pose; (*2*) improve performance; (*3*) reduce the required processing power; and (*4*) facilitate reuse in other real applications. This formulation not only avoids the *Gimbal Lock*, but also allows a better performance in numerical implementation thanks to the linearity of quaternion algebra [42], which facilitates its implementation and improves its accuracy. For example, while Miletović et al. [41] indicate a high accuracy of the estimated states with an amplitude mismatch in the range of several tenths of a millimeter, our quaternion-based method has errors in the millimeter unit.

Regarding the processing power, some works have optimized the Kalman filter by using FPGAs. Lee et al. [43] were the first to propose a reconfigurable FPGA-based Kalman filtering co-processor for multi-target tracking radar systems. Replacing the traditional software with a hardware-based algorithm, the performance improved by two to three orders of magnitude, achieving a cycle time of 1.824 microseconds. Bonato et al. [44] and Guo et al. [45] ran a C-based EKF algorithm on a Nios II processor embedded in an FPGA device. Both approaches adopted customized instructions and algorithm optimization to make them more efficient. Our work uses the FPGA approach to improve performance and accurately estimate the pose in real-time (i.e., support online applications). Furthermore, in addition to using a conventional webcam, our approach is concerned with using a commercial processor easily found in inexpensive embedded systems available on the market (i.e., ARM Cortex-M3) [46, 47], facilitating its reuse through design methodologies in other real systems [48]. All of these advantages are summarized in Table 1, which clearly indicates the positioning of our work in relation to the state-of-the-art.

3 Background

This section presents the main concepts and definitions used throughout this work as a basis for proposing the EKF algorithm to estimate the pose of 6–DOF systems. First, essential mathematical properties associated with quaternion parameterization are presented. Next, we define

Table 1 State-of-the-Art summary, highlighting the advantages of the proposed work

Work	Sensing technology	Stewart Platform	Application processing	KF-based approach	FPGA-based approach
This work	Vision-sensor based	yes	online	yes	yes
[36, 38–40]	Vision-sensor based	no	online	yes	no
[37]	Vision-sensor based	no	offline	yes	no
[34]	Vision-based	yes	online	no	no
[29]	Vision-based	no	online	no	no
[35]	Vision-based	no	offline	yes	no
[33]	Vision-based	no	offline	no	no
[41]	Sensor-based	yes	online	yes	no
[25, 32]	Sensor-based	yes	online	no	no
[30, 31]	Sensor-based	yes	offline	no	no
[44, 45]	–	no	online	yes	yes
[43]	–	no	–	yes	yes

the mathematical representation of the sensors and describe the Extended Kalman Filter. Finally, the geometry and mathematical relations of the Stewart platform dynamics are formulated.

3.1 Quaternions

Quaternions can be interpreted as a mathematical tool used to model rotations in 3D space as an alternative to Euler's angles [15]. The great advantage is the ease of interpolation between two quaternions, which is extremely useful for detecting smooth camera movements. Its vector form $\boldsymbol{q} \in \mathbb{R}^4$ can be represented as:

$$\boldsymbol{q} = \begin{bmatrix} \eta \\ \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \end{bmatrix} = \begin{bmatrix} \eta \\ \boldsymbol{\varepsilon} \end{bmatrix} = \begin{bmatrix} cos(\frac{\theta}{2}) \\ \boldsymbol{r} sin(\frac{\theta}{2}) \end{bmatrix}. \quad (1)$$

Belonging to the hypercomplex number set, we have $\eta \in \mathbb{R}$ as the real component of quaternions and $\boldsymbol{\varepsilon} \in \mathbb{R}^3$ as the imaginary component [15]. Physically, Eq. 1 represents a rotation θ in the $\boldsymbol{r} \in \mathbb{R}^3$ direction. Therefore, based on the quaternions representation, the rotation matrix, i.e., $R(\boldsymbol{q}) \in \mathbb{R}^{3\times3}$ can be write as:

$$R(\boldsymbol{q}) = I_3 + 2\eta S(\boldsymbol{\varepsilon}) + 2S^2(\boldsymbol{\varepsilon}), \quad (2)$$

where $I_3 \in \mathbb{R}^3$ is an identity matrix and operator $S(\cdot) : \mathbb{R}^3 \to \mathbb{R}^{3\times3}$ is hereby expressed by:

$$S(\boldsymbol{\varepsilon}) = \begin{bmatrix} 0 & -\varepsilon_3 & \varepsilon_2 \\ \varepsilon_3 & 0 & -\varepsilon_1 \\ -\varepsilon_2 & \varepsilon_1 & 0 \end{bmatrix}, \quad (3)$$

where ε_i is the i-th component of $\boldsymbol{\varepsilon}$.

3.2 Inertial Measurement Sensor Models

The gyroscope and accelerometer are inertial measurement sensors capable of determining the angular velocity and acceleration of a rigid body relative to the sensor's local coordinate system. Since the local sensor coordinate is normally different from the global coordinate, it is required to have a model suitable of relating the angular velocity ($\boldsymbol{\omega}_G$), measured by the gyroscope sensor, to the angular velocity ($\boldsymbol{\omega}$) of the center rigid body, which is based on global coordinates. Considering this representation and to obtain information about the system orientation, the gyroscope model is represented as follows:

$$\boldsymbol{\omega}_G = R_{imu}\boldsymbol{\omega} + \boldsymbol{b}_G + \boldsymbol{\delta}_G, \quad (4)$$

where R_{imu} is a rotation matrix which describes the IMU orientation with respect to the rigid body orientation, $\boldsymbol{b}_G \in \mathbb{R}^3$ represents the sensor offset, and finally, $\boldsymbol{\delta}_G$ is a stochastic process representing the Gaussian white noise of the sensor, with zero mean and Q_G covariance.

One way to obtain information about the translation behavior of a system is to use measurements from an accelerometer. Normally, the origin of the sensor is not located in the system's origin, thus centrifugal force and Coriolis force must be considered [49]. Therefore, the mathematical representation of the local acceleration coordinate $\boldsymbol{a}_A$ is given by:

$$\boldsymbol{a}_A = R_{imu}R^T(\boldsymbol{q})((\boldsymbol{a}+\boldsymbol{g})+H(\boldsymbol{\omega},\boldsymbol{a})\boldsymbol{O}_{imu})+\boldsymbol{b}_A+\boldsymbol{\delta}_A, \quad (5)$$

where $R_{imu}R^T(\boldsymbol{q})$ is the rotation matrix responsible for transforming the measurement orientation, $\boldsymbol{a}$ is the acceleration in the center rigid body, and $\boldsymbol{g}$ is the gravity vector. In addition, $H(\boldsymbol{\omega}, \boldsymbol{\alpha})$ function represents the effects of inertial forces, given by:

$$H(\boldsymbol{\omega}, \boldsymbol{\alpha}) = S(\boldsymbol{\omega})^2 + S(\boldsymbol{\alpha}), \quad (6)$$

where $\boldsymbol{O}_{imu} \in \mathbb{R}^3$ represents the sensor offset from the origin. Lastly, $\boldsymbol{b}_A \in \mathbb{R}^3$ represents the sensor offset and $\boldsymbol{\delta}_A$ is a stochastic process representing the Gaussian white noise of the sensor, with zero mean and Q_A covariance.

3.3 Camera Model

The ideal perspective model of a pinhole camera describes the relationship between a point $\boldsymbol{\xi}_i \in \mathbb{R}^3$ referenced in the global frame, and its projection in pixels in the image plane $\boldsymbol{m}_i = \begin{bmatrix} x_{img,i} & y_{img,i} \end{bmatrix}^T$, where i is the marker number. Therefore, to obtain a mathematical modeling of the camera, its intrinsic and extrinsic parameters must be considered. Vasquez et al. [50] showed that the camera's model with orientation $\boldsymbol{q}$ and position $\boldsymbol{p} \in \mathbb{R}^3$ can be represented as follows:

$$0 = S(\bar{\boldsymbol{m}}_i) K_{int} R_{cam} R^T(\boldsymbol{q})(\boldsymbol{\xi}_i - \boldsymbol{p}) + \boldsymbol{\delta}_{cam}, \tag{7}$$

where $\bar{\boldsymbol{m}}_i = \begin{bmatrix} \boldsymbol{m}_i & 1 \end{bmatrix}^T$ is the marker position in the image frame of the point $\boldsymbol{\xi}_i$ in pixels, R_{cam} is a rotation matrix which describes the camera orientation with respect to the rigid body orientation, $\boldsymbol{\delta}_{cam}$ represents the Gaussian white noise of the camera with zero mean and Q_{cam} covariance and K_{int} is the matrix with the intrinsic parameters, given by:

$$K_{int} = \begin{bmatrix} fs_x & fs_\theta & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{8}$$

where (o_x, o_y) are the coordinate from center position in pixel of image plane, s_x and s_y represents the unit measurements in pixels of horizontal and vertical direction, f is the focal length and fs_θ is the skew of the pixel.

3.4 Extended Kalman Filter

The Extended Kalman Filter is commonly used for sensor fusion and state observation. The great advantage of the EKF is that this approach can be applied to a broader class of systems, as nonlinear systems, if compared to the classical version of the filter. In this regard, we can describe the state dynamics of a general discrete time-variant nonlinear system as:

$$\begin{aligned} \boldsymbol{x}_k &= f(\boldsymbol{u}_k, \boldsymbol{x}_{k-1}) + \boldsymbol{\epsilon}_k, \\ z_k &= h(\boldsymbol{x}_k) + \boldsymbol{\delta}_k, \end{aligned} \tag{9}$$

where $f(\cdot)$ defines the system dynamics, $\boldsymbol{x}_k \in \mathbb{R}^n$ defines the system state vector, $\boldsymbol{u}_k \in \mathbb{R}^m$ is the control vector, $z_k \in \mathbb{R}^r$ is the observation vector, and $\boldsymbol{\epsilon}_k, \boldsymbol{\delta}_k$ are zero-mean Gaussian random vectors that model the uncertainty introduced by the state transition equation and measurement processes, respectively [51].

Table 2 EKF Algorithm: Prediction (lines 1 and 2) and Update (lines 3-5)

Prediction	Update
1: $\bar{x}_k = f(\boldsymbol{u}_k, \boldsymbol{x}_{k-1})$	3: $K_k = \bar{P}_k H_k^T (H_k \bar{P}_k H_k^T + Q_z)^{-1}$
2: $\bar{P}_k = F_k P_{k-1} F_k^T + Q_x$	4: $\hat{\boldsymbol{x}}_{k+1} = \bar{\boldsymbol{x}}_k + K_k(z_k - h(\bar{\boldsymbol{x}}_k))$
	5: $\hat{P}_{k+1} = (I - K_k H_k)\bar{P}_k$

The EKF has two distinct steps, as previously mentioned, where the prediction step computes the system states $\bar{\boldsymbol{x}}_k$ and covariance $\bar{P}_k \in \mathbb{R}^{n \times n}$ based on the *a priori* system states $\bar{\boldsymbol{x}}_{k-1}$ and covariance $\bar{P}_{k-1}$, lines 1 and 2 from Table 2. Afterward, the update step computes the K_k Kalman gain and the *a posteriori* system state $\hat{\boldsymbol{x}}_{k+1}$ and covariance $\hat{P}_{k+1}$ with Q_x model covariance, Q_z sensor covariance and H_k Jacobian matrix, lines 3, 4 and 5 from Table 2.

Taking the above-mentioned models and the EKF algorithm one can conclude that the developed approach can be implemented to any systems with a maximum of 6–DOF. To validate our approach in a real-time application, a Stewart platform has been considered.

3.5 Stewart Platform

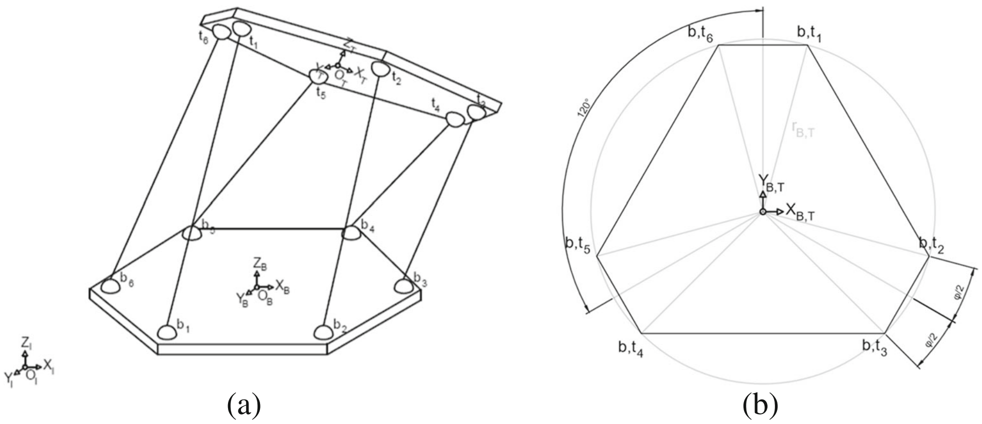
The Stewart platform consists of a fixed and a mobile platform, connected by six linear actuators, which makes this system capable of moving in all 6–DOF. According to Dasgupta et al. [52], the Stewart platform, as a parallel-type robot, presents several advantages over robots with series structures, such as mechanical rigidity; high precision and low torque demand in relation to the load [53]. Furthermore, the type of joint used does not introduce bending efforts on them, as it only imposes displacement restrictions. This, added to the fact that the weight of the load is distributed, allows minimizing the structural rigidity of the actuators, reducing the power and the size required for them.

3.5.1 Stewart Platform Geometry

Figure 2a shows three coordinate systems: O_I is the global or interest coordinate system; O_B is the reference system located in the center at the bottom of the platform and O_T is the coordinate system located at the top of the platform.

The manipulator has the actuator $i = (1...6)$ and the bottom joints b_i and top joints t_i, arranged in pairs and separated from each other at 120º, belong to a vertex of a hexagonal enclose in a circle of bottom radius r_B and top radius r_T, as seen in Fig. 2b [54]. Finally, φ_B and φ_T also define the distance between the joints, as well as the size of the platform edge.

Next, the position of the b_i and t_i joints are defined using the vectors $\boldsymbol{B}_i \in \mathbb{R}^3$ for the bottom of the platform and



(a) (b)

Fig. 2 **a** Stewart Platform Coordinate System, and **b** Geometric Representation. Adapted from [54]

$\boldsymbol{T}_i \in \mathbb{R}^3$ for the top of the platform in relation to their respective local coordinate systems, which are expressed as follows:

$$\boldsymbol{B}_i = \begin{bmatrix} r_B \cos(\lambda_i) \\ r_B \sin(\lambda_i) \\ 0 \end{bmatrix}, \quad \boldsymbol{T}_i = \begin{bmatrix} r_T \cos(\psi_i) \\ r_T \sin(\psi_i) \\ 0 \end{bmatrix}, \tag{10}$$

where

$$\lambda_i = \begin{cases} \frac{i\pi}{3} - \frac{\varphi_B}{2} & i = (1,3,5) \\ \lambda_{i-1} - \varphi_B & i = (2,4,6) \end{cases}, \quad \psi_i = \begin{cases} \frac{i\pi}{3} - \frac{\varphi_T}{2} & i = (1,3,5) \\ \psi_{i-1} - \varphi_T & i = (2,4,6) \end{cases}. \tag{11}$$

3.5.2 Stewart Platform Dynamics

Lagrangian formalism, based on system energy, and Newton-Euler formalism, which uses the forces surrounding plant movement, are commonly used to describe the dynamics of the Stewart platform. However, this work proposes the use of quaternion-based formalism. This makes our approach use a generic description of a 3D rigid body with respect to the coordinate system whose origin coincides with the body's center of mass. Therefore, based on the previous section and the equations presented by Markley et al. [55], dynamical equations of the position and orientation of the center of the top of the platform can be expressed as:

$$\begin{bmatrix} \dot{\boldsymbol{q}} \\ \dot{\boldsymbol{\omega}} \\ \dot{\boldsymbol{p}} \\ \dot{\boldsymbol{v}} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\begin{bmatrix} -\boldsymbol{\varepsilon}^T \\ \eta I_3 + S(\boldsymbol{\varepsilon}) \end{bmatrix}\boldsymbol{\omega} \\ I_m^{-1}(\boldsymbol{u}_\tau + \boldsymbol{\tau}_{ext} - S(\boldsymbol{\omega}) I_m \boldsymbol{\omega}) \\ \boldsymbol{v} \\ M^{-1}(\boldsymbol{u}_F + \boldsymbol{F}_{ext}) + \boldsymbol{g} \end{bmatrix}, \tag{12}$$

where $\boldsymbol{u}_\tau \in \mathbb{R}^3$ and $\boldsymbol{\tau}_{ext} \in \mathbb{R}^3$ are the input and disturbance torques related to the local coordinate system, $\boldsymbol{u}_F \in \mathbb{R}^3$ and $\boldsymbol{F}_{ext} \in \mathbb{R}^3$ are the input and disturbance forces related to the global coordinate system and $\boldsymbol{g}$ is the gravity vector.

Analyzing Eq. 12, it is noticed that the first two lines of the matrix are related to the rotational movement of the top of the platform, which is given by:

$$\begin{aligned} \dot{\boldsymbol{q}}(\eta, \boldsymbol{\varepsilon}, \boldsymbol{\omega}) &= \frac{1}{2}\begin{bmatrix} -\boldsymbol{\varepsilon}^T \\ \eta I_3 + S(\boldsymbol{\varepsilon}) \end{bmatrix}\boldsymbol{\omega}, \\ \boldsymbol{\tau}(\boldsymbol{\omega}, \dot{\boldsymbol{\omega}}) &= I_m \dot{\boldsymbol{\omega}} + S(\boldsymbol{\omega}) I_m \boldsymbol{\omega}, \end{aligned} \tag{13}$$

where $\boldsymbol{\tau} \in \mathbb{R}^3$ is the torque applied to the body, $I_m \in \mathbb{R}^3$ represents the inertia matrix and $\dot{()}$ represents the time derivatives. In the same sense, the last two lines of Eq. 12, related to the translation dynamic equations at the top of the platform, are defined as:

$$\begin{aligned} \dot{\boldsymbol{p}}(\boldsymbol{v}) &= \boldsymbol{v}, \\ \boldsymbol{F}(\dot{\boldsymbol{v}}) &= M\dot{\boldsymbol{v}}, \end{aligned} \tag{14}$$

where $\dot{\boldsymbol{p}} = \begin{bmatrix} \dot{p}_x & \dot{p}_y & \dot{p}_z \end{bmatrix}^T \in \mathbb{R}^3$ is the velocity vector of the platform top with respect to the global coordinate system, with $\dot{p}_x$, $\dot{p}_y$ and $\dot{p}_z$ related to the x-, y- and z-$axis$, respectively. $M \in \mathbb{R}^{3\times3}$ is the body mass matrix, $\boldsymbol{F}(\dot{\boldsymbol{v}}) \in \mathbb{R}^3$ is the inertial force matrix and $\dot{\boldsymbol{v}} \in \mathbb{R}^3$ is the acceleration vector, the latter two being also represented in the global coordinate system.

4 Proposed Pose Estimation Algorithm

4.1 Theoretical Pose Estimation

Based on the preliminary concepts shown in Section 3, this section presents the Extended Kalman Filter design in detail, which is divided into *prediction* and *update* steps.

In order to embedded the system into an FPGA, the discrete representation of the model is needed. Thus, through Euler's discretization $\dot{\boldsymbol{x}}_k \approx (\boldsymbol{x}_{k+1} - \boldsymbol{x}_k)/T_s$, for a time interval T_s, the system model can be described as follows:

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k) + \boldsymbol{\epsilon}_k, \tag{15}$$

where the index k indicates the discrete representation.

In the *prediction* step, the system state vector is given by the orientation, angular velocity, angular acceleration, position, linear velocity, linear acceleration, gyroscope bias and accelerometer bias, which are respectively represented by the following state vector:

$$\hat{\boldsymbol{x}}_k := \left[\hat{\boldsymbol{q}}_k^T \ \hat{\boldsymbol{\omega}}_k^T \ \hat{\boldsymbol{\alpha}}_k^T \ \hat{\boldsymbol{p}}_k^T \ \hat{\boldsymbol{v}}_k^T \ \hat{\boldsymbol{a}}_k^T \ \hat{\boldsymbol{b}}_{G,k}^T \ \hat{\boldsymbol{b}}_{A,k}^T\right]^T \in \mathbb{R}^{25}, \tag{16}$$

whereby

$$f(\hat{\boldsymbol{x}}_k) = \hat{\boldsymbol{x}}_k + T_s \begin{bmatrix} \mathbb{F}(\hat{\boldsymbol{\omega}}_k)\hat{\boldsymbol{q}}_k \\ \hat{\boldsymbol{\alpha}}_k \\ 0_{3\times 1} \\ \hat{\boldsymbol{v}}_k \\ \hat{\boldsymbol{a}}_k \\ 0_{9\times 1} \end{bmatrix}, \tag{17}$$

and

$$\mathbb{F}(\hat{\boldsymbol{\omega}}_k) = \frac{1}{2}\begin{bmatrix} 0 & -\hat{\boldsymbol{\omega}}_k^T \\ \hat{\boldsymbol{\omega}}_k & -S(\hat{\boldsymbol{\omega}}_k) \end{bmatrix}. \tag{18}$$

In this regard, the vector $\boldsymbol{\epsilon}_k \in \mathbb{R}^{25}$ is the stochastic process representing the uncertainty introduced in the state transition equation. This uncertainty is related to the covariance matrix Q_x, which is a parameter that must be adjusted.

The next step is the linearization of the Eq. 15, which is given by the following Jacobian matrix:

$$F(\hat{\boldsymbol{x}}_k) = \frac{\partial f(\hat{\boldsymbol{x}}_k)}{\partial \hat{\boldsymbol{x}}_k} = I_{25} + T_s \, diag\{\Phi(\hat{\boldsymbol{x}}_k), \Psi, 0_{6\times 6}\}, \tag{19}$$

where $diag\{\cdot\}$ means the diagonal of a matrix,

$$\Phi(\hat{\boldsymbol{x}}_k) = \begin{bmatrix} \mathbb{F}(\hat{\boldsymbol{\omega}}_k) & \mathbb{G}(\hat{\boldsymbol{q}}_k) & 0_{4\times 3} \\ 0_{3\times 4} & 0_3 & I_3 \\ 0_{3\times 4} & 0_3 & 0_3 \end{bmatrix}, \ \Psi = \begin{bmatrix} 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & I_3 \\ 0_3 & 0_3 & 0_3 \end{bmatrix} \tag{20}$$

and

$$\mathbb{G}(\hat{\boldsymbol{q}}_k) = \frac{1}{2}\begin{bmatrix} -\hat{\boldsymbol{\varepsilon}}_k^T \\ \hat{\eta}_k I + S(\hat{\boldsymbol{\varepsilon}}_k) \end{bmatrix}. \tag{21}$$

In the *update* step, we first define the measurement vector of the z_k sensor. Note that in our application we selected to use four markers with different colors. Thus, the vector z_k, using the Eqs. 4, 5 and 7, is defined as:

$$z_k = \begin{bmatrix} \boldsymbol{\omega}_{G,k} \\ \boldsymbol{a}_{A,k} \\ 0_{8\times 1} \end{bmatrix} \in \mathbb{R}^{14}. \tag{22}$$

The nonlinear output equation described as z_k will be reconstructed from $\hat{\boldsymbol{x}}_k$, which is represented by:

$$z_k = \boldsymbol{h}(\hat{\boldsymbol{x}}_k) + \boldsymbol{\delta}_k, \tag{23}$$

where we can define $\boldsymbol{h}_k(\hat{\boldsymbol{x}}_k)$ also from the Eqs. 4, 5 and 7,

$$\boldsymbol{h}_k(\hat{\boldsymbol{x}}_k) = \begin{bmatrix} R_{imu}\hat{\boldsymbol{\omega}}_k + \hat{\boldsymbol{b}}_{G,k} \\ R_{imu}R^T(\hat{\boldsymbol{q}}_k)((\hat{\boldsymbol{a}}_k + \boldsymbol{g}) + H(\hat{\boldsymbol{\omega}}_k, \hat{\boldsymbol{\alpha}}_k)\boldsymbol{O}_{imu} + \hat{\boldsymbol{b}}_{A,k} \\ CC\cdot S(\bar{\boldsymbol{m}}_1)K_{int}R_{cam}R^T(\hat{\boldsymbol{q}}_k)(\boldsymbol{\xi}_1 - \hat{\boldsymbol{p}}_k) \\ CC\cdot S(\bar{\boldsymbol{m}}_2)K_{int}R_{cam}R^T(\hat{\boldsymbol{q}}_k)(\boldsymbol{\xi}_2 - \hat{\boldsymbol{p}}_k) \\ CC\cdot S(\bar{\boldsymbol{m}}_3)K_{int}R_{cam}R^T(\hat{\boldsymbol{q}}_k)(\boldsymbol{\xi}_3 - \hat{\boldsymbol{p}}_k) \\ CC\cdot S(\bar{\boldsymbol{m}}_4)K_{int}R_{cam}R^T(\hat{\boldsymbol{q}}_k)(\boldsymbol{\xi}_4 - \hat{\boldsymbol{p}}_k) \end{bmatrix} \tag{24}$$

and we define the stochastic variable $\boldsymbol{\delta}_k$ as

$$\boldsymbol{\delta}_k = \left[\boldsymbol{\delta}_{G,k}^T \ \boldsymbol{\delta}_{A,k}^T \ \boldsymbol{\delta}_{cam,k}^T\right]^T, \tag{25}$$

where $CC = \left[I_2 \ 0_{2\times 1}\right]$.

Therefore, the Q_z covariance matrix for all the uncertainties of the sensor measurements is composed of the covariance matrices of each sensor, forming a diagonal block in the following form:

$$Q_z = diag\{Q_G, Q_A, Q_{cam}\}. \tag{26}$$

The Jacobian matrix $H(\hat{\boldsymbol{x}}_k) \in \mathbb{R}^{14\times 25}$ is formed by the partial derivatives of $\boldsymbol{h}(\hat{\boldsymbol{x}}_{\boldsymbol{k}})$ in relation to $\hat{\boldsymbol{x}}_k$ and is defined as:

$$H_k(\hat{\boldsymbol{x}}_k) := \frac{\partial \boldsymbol{h}(\hat{\boldsymbol{x}}_k)}{\partial \hat{\boldsymbol{x}}_k} = \left[H_k^{\hat{\boldsymbol{q}}_k} \ H_k^{\hat{\boldsymbol{\omega}}_k,\hat{\boldsymbol{\alpha}}_k} \ H_k^{\hat{\boldsymbol{p}}_k} \ H_k^{\hat{\boldsymbol{v}}_k,\hat{\boldsymbol{a}}_k} \ H_k^{\hat{\boldsymbol{b}}_k}\right], \tag{27}$$

where $H_k^{\hat{\boldsymbol{q}}_k}$, $H_k^{\hat{\boldsymbol{\omega}}_k,\hat{\boldsymbol{\alpha}}_k}$, $H_k^{\hat{\boldsymbol{p}}_k}$, $H_k^{\hat{\boldsymbol{v}}_k,\hat{\boldsymbol{a}}_k}$ and $H_k^{\hat{\boldsymbol{b}}_k}$ are the Jacobians related with the orientation, angular velocity and acceleration, position, linear velocity and acceleration and bias dynamics, respectively. The result of partial derivatives are:

$$H_k^{\hat{\boldsymbol{q}}_k} := \begin{bmatrix} 0_{3\times 4} \\ R_{imu}\mathbb{J}(\hat{\boldsymbol{q}}_k, \hat{\boldsymbol{a}}_k + \boldsymbol{g}) \\ CC\cdot S(\bar{\boldsymbol{m}}_1)K_{int}R_{cam}\mathbb{J}(\hat{\boldsymbol{q}}_k, \boldsymbol{\xi}_1 - \hat{\boldsymbol{p}}_k) \\ CC\cdot S(\bar{\boldsymbol{m}}_2)K_{int}R_{cam}\mathbb{J}(\hat{\boldsymbol{q}}_k, \boldsymbol{\xi}_2 - \hat{\boldsymbol{p}}_k) \\ CC\cdot S(\bar{\boldsymbol{m}}_3)K_{int}R_{cam}\mathbb{J}(\hat{\boldsymbol{q}}_k, \boldsymbol{\xi}_3 - \hat{\boldsymbol{p}}_k) \\ CC\cdot S(\bar{\boldsymbol{m}}_4)K_{int}R_{cam}\mathbb{J}(\hat{\boldsymbol{q}}_k, \boldsymbol{\xi}_4 - \hat{\boldsymbol{p}}_k) \end{bmatrix}, \tag{28}$$

$$H_k^{\hat{\boldsymbol{\omega}}_k,\hat{\boldsymbol{\alpha}}_k} := \begin{bmatrix} R_{imu}R(\hat{\boldsymbol{q}}_k) & 0_{3\times 3} \\ -R_{imu}R(\hat{\boldsymbol{q}}_k)\mathbb{W}(\hat{\boldsymbol{\omega}}_k, \boldsymbol{O}_{imu}) & -R_{imu}R(\hat{\boldsymbol{q}}_k)S(\boldsymbol{O}_{imu}) \\ 0_{8\times 3} & 0_{8\times 3} \end{bmatrix}, \tag{29}$$

$$H_k^{\hat{\boldsymbol{p}}_k} := \begin{bmatrix} 0_{6\times 3} \\ -CC\cdot S(\bar{\boldsymbol{m}}_1)K_{int}R_{cam}R(\hat{\boldsymbol{q}}_k) \\ -CC\cdot S(\bar{\boldsymbol{m}}_2)K_{int}R_{cam}R(\hat{\boldsymbol{q}}_k) \\ -CC\cdot S(\bar{\boldsymbol{m}}_3)K_{int}R_{cam}R(\hat{\boldsymbol{q}}_k) \\ -CC\cdot S(\bar{\boldsymbol{m}}_4)K_{int}R_{cam}R(\hat{\boldsymbol{q}}_k) \end{bmatrix}, \tag{30}$$

$$H_k^{\hat{\boldsymbol{v}}_k,\hat{\boldsymbol{a}}_k} := \begin{bmatrix} 0_3 & 0_3 \\ 0_3 & R_{imu}R(\hat{\boldsymbol{q}}_k) \\ 0_{8\times 3} & 0_{8\times 3} \end{bmatrix} \tag{31}$$

and $H_k^{\hat{b}_k} = \left[I_6 \; 0_{6\times 8}\right]^T$. The auxiliary functions $\mathbb{J}(\boldsymbol{q},\boldsymbol{u})$ and $\mathbb{W}(\boldsymbol{y},\boldsymbol{u}) \; \forall \boldsymbol{y},\boldsymbol{u} \in \mathbb{R}^3$ are defined as:

$$\mathbb{J}(\boldsymbol{q},\boldsymbol{u}) := \frac{\partial R(\boldsymbol{q})\boldsymbol{u}}{\partial \boldsymbol{q}} = \frac{\partial (I_3 - 2\eta S(\boldsymbol{\varepsilon}) + 2S^2(\boldsymbol{\varepsilon}))\boldsymbol{u}}{\partial \begin{bmatrix}\eta \\ \boldsymbol{\varepsilon}\end{bmatrix}}$$
$$= 2\left[-S(\boldsymbol{\varepsilon})\boldsymbol{u}S(\boldsymbol{u})\eta + \mathbb{W}(\boldsymbol{\varepsilon},\boldsymbol{u})\right] \qquad (32)$$

and

$$\mathbb{W}(\boldsymbol{y},\boldsymbol{u}) := \frac{\partial S^2(\boldsymbol{y})\boldsymbol{u}}{\partial \boldsymbol{y}} = \boldsymbol{y}\boldsymbol{u}^T - 2\boldsymbol{u}\boldsymbol{y}^T + \boldsymbol{y}^T\boldsymbol{u}I_3. \qquad (33)$$

From the definitions formulated in this section, we were able to develop the EKF algorithm using the equations shown in Table 2. Finally, to guarantee the consistency of the proposed model, after the prediction and update steps, it is mandatory to normalize the estimated quaternion $\boldsymbol{q}_k$ [56], that is, we have that $\boldsymbol{q}_k = \boldsymbol{q}_k/||\boldsymbol{q}_k||$ where $||\cdot||$ represents the Euclidean norm.

4.2 Proposed Embedded Software

Due to the growing environment diversity in which 6–DOF systems are employed, we necessarily require to develop new cost-effective approaches that improve performance and also reduce costs for the industry. In this sense, besides demonstrating mathematically how we can improve pose estimation with EKF based on IMU and visual marker measurements, as shown in Section 4.1. This work proposed its implementation through an embedded software that fits into an FPGA device with commercial processors, facilitating the reproduction and use of our approach in the most diverse 6–DOF systems.

Table 3 Developed libraries to simplify the implementation of the pose estimation algorithm

Libraries	Number of Functions	Line of Code (LOC)
EKF	2	496
Quaternion	7	141
Communication Protocol	12	420
Matrix Functions	24	406

Figure 3 shows the adopted three-phase development flow. The first phase was devoted to the development of the algorithm. Initially, a MATLAB application was developed to model the theoretical pose estimation presented in Section 4.1. This is our golden reference for ensuring the correctness of our mathematical expressions, as well as used to validate our algorithm.

Next, the algorithm was described in C language to calculate the pose estimates, as shown in Table 2. In this sense, libraries were developed to handle operations with quaternions and matrices, e.g., functions to calculate the inverse matrix, multiplications, divisions, Jacobian matrix, Euclidean norm of the quaternion, derivative, cross product, among other functions. Table 3 shows the number of functions and lines implemented for each application library.

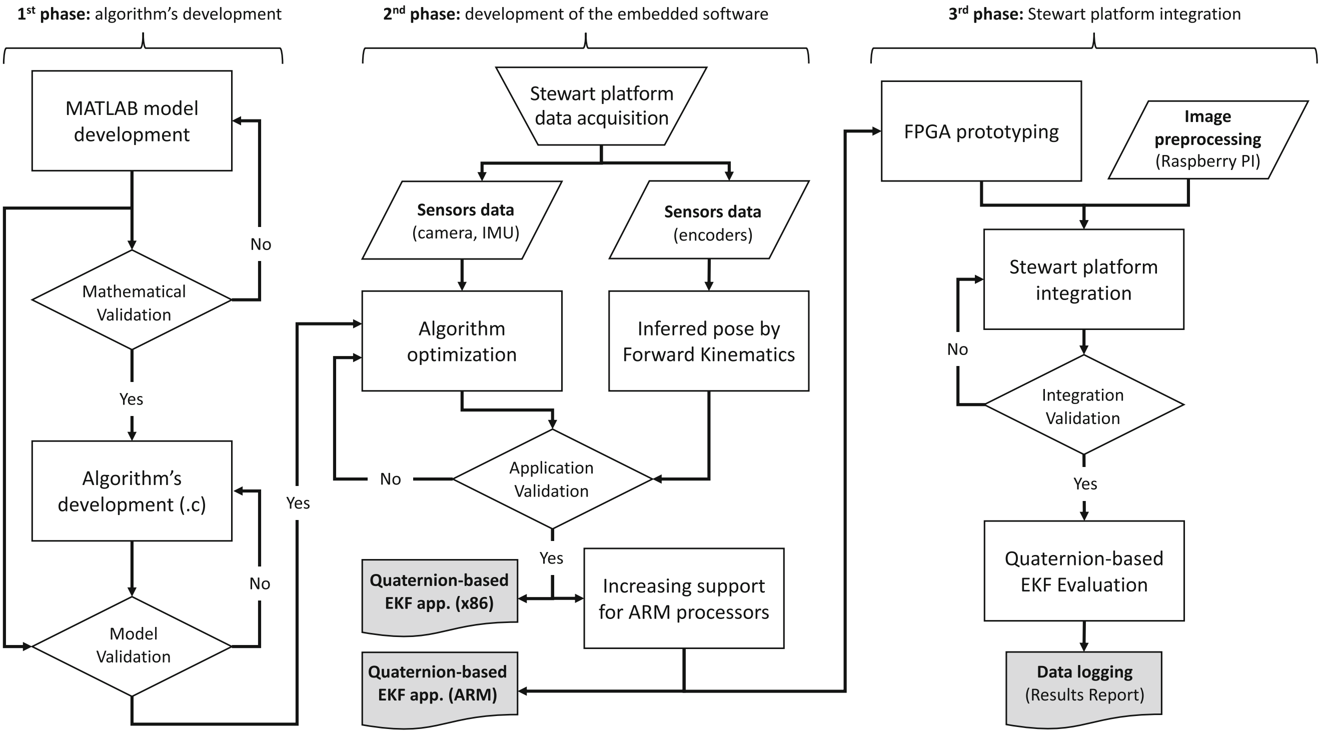

1st phase: algorithm's development
2nd phase: development of the embedded software
3rd phase: Stewart platform integration
MATLAB model development
Mathematical Validation
No
Yes
Algorithm's development (.c)
No
Model Validation
Yes
Stewart platform data acquisition
Sensors data (camera, IMU)
Sensors data (encoders)
Algorithm optimization
Inferred pose by Forward Kinematics
No
Application Validation
Yes
Quaternion-based EKF app. (x86)
Increasing support for ARM processors
Quaternion-based EKF app. (ARM)
FPGA prototyping
Image preprocessing (Raspberry PI)
Stewart platform integration
No
Integration Validation
Yes
Quaternion-based EKF Evaluation
Data logging (Results Report)

Fig. 3 Development flow with 3 distinct phases.

In the second phase, after validating the algorithm, we evaluated our application on a standard desktop PC based on the x86 processor. This step allowed us to ensure the correct implementation of our pose estimator and also aims to facilitate the dissemination of our application to be applied to other 6–DOF systems. At this stage, all input data from the sensors (i.e., encoders, IMU and camera) have been acquired and preprocessed offline, making it possible to optimize the algorithm, i.e., adjust the covariances and gains of the EKF. To validate our application, the estimated states were compared with the real data, calculated from the encoders using forward kinematics. After validation, we increased support for ARM processors as well, to cover the vast majority of devices found in the industry and prepare it to be embedded into an FPGA device, either as embedded software or through the use of high-level synthesis [57].

The third phase addresses the integration of the FPGA with the Stewart platform to support online applications. First, we prototype the application into the FPGA. Then, image preprocessing is done by a Raspberry PI 3 Model B, where data extracted from each frame are grouped with IMU data and sent to the FPGA through a developed protocol. The goal of dividing the problem into smaller tasks is to make implementation cheaper or even to take advantage of multitasking support in a multi-core processor, since this design choice helps such architectures to be more power-efficient [58].

Once the integration is validated, the experiments can be started. After every rehearsal, which consists of several frames, a data logging is created with the pose estimation calculated by the FPGA with all sensor data (i.e., IMU and camera). Then, this filter is evaluated by comparing the estimated data with the actual data calculated from the Stewart platform, showing the effectiveness of our approach.

4.3 Data Acquisition

This section is dedicated to detailing the data acquisition process shown in the second phase of Fig. 3. The Stewart platform used is equipped with an IMU and a camera to estimate the pose. Regarding the sensing part, the IMU contains a gyroscope, an accelerometer and a magnetometer. The system orientation is obtained by integrating the angular velocity measurements inferred by the gyroscope. In contrast, acceleration and linear position is obtained by integrating the signal from the accelerometer. Figure 4a represents the DOF measured by the IMU.

On the other hand, the vision part is important due to the large amount of information that an image provides. However, as seen in the literature review, the disadvantage is the demand for a huge computational effort. Currently, the main and most used tool for dealing with images is the OpenCV library, which has algorithms optimized for different applications, such as face recognition, object identification, 3D model extraction, filter applications and others.

To reduce the computing effort, our approach used a Logitech C270 webcam with HD resolution to have the lowest possible covariance in measurements. In addition, markers with different colors and simple shapes were used to reduce the computational effort involved. Figure 4b shows how the markers were organized, where three markers are on the same plane with different positions. Furthermore, to enrich the image with information, another marker was allocated with a different position and plane.

The procedure for acquiring the marker positions is as follows: after making the image acquisition, a conversion from RGB to HSV is performed; then the marker of interest is selected according to the hue, saturation and brightness values; after, the contour detection can be performed. If a marker is in the camera's field of view, it is then possible to obtain the marker's coordinate through the image's invariant moment properties, as formulated by Teague [59]. In this work, the calculated centroid values are the positions of the markers in pixels in the image plane.

To infer the pose through forward kinematics, we use encoders arranged next to the DC motors, which are responsible for driving the actuators. Figure 5 shows this

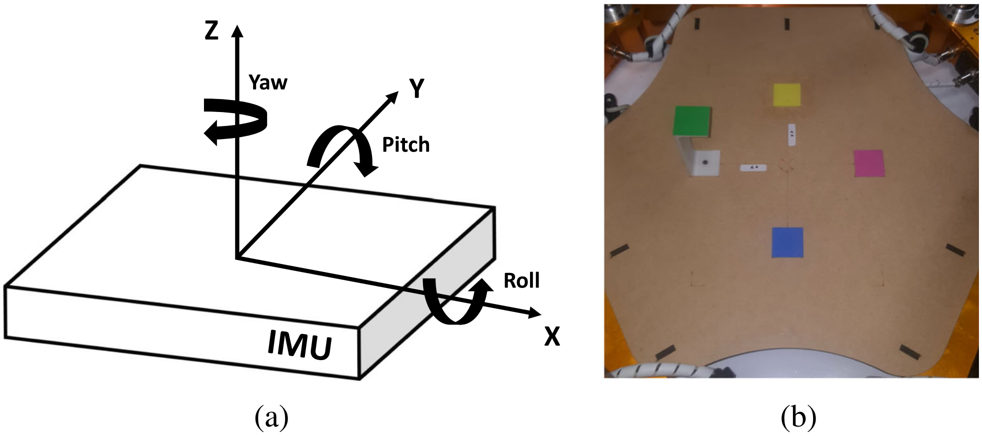


Fig. 4 **a** Graphical representation of the DOF measured by the IMU, and **b** Markers organization to reduce the computational effort to estimate the pose of a Stewart platform

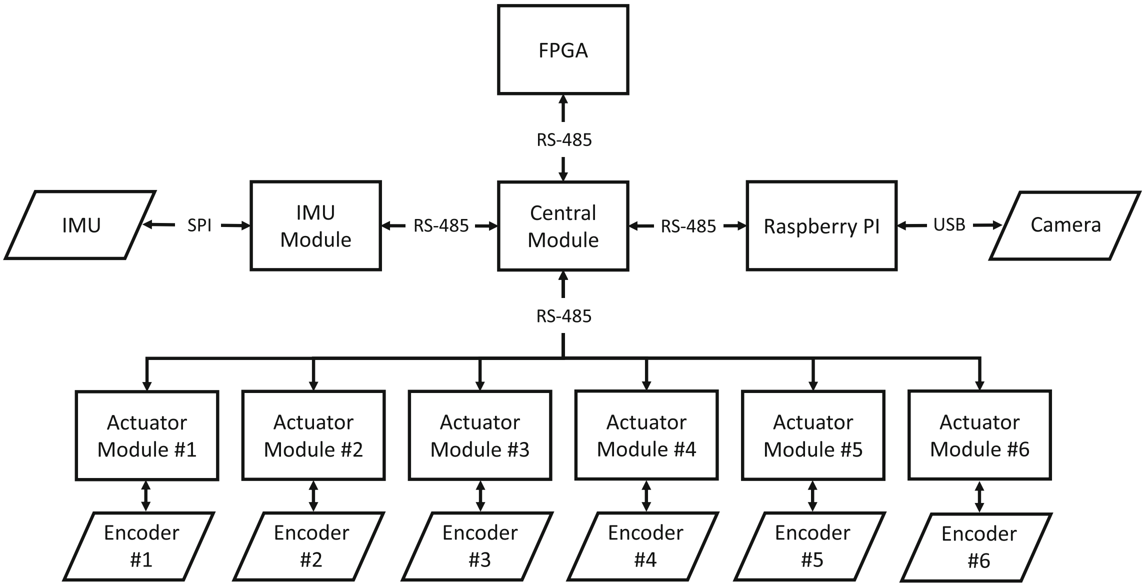


Fig. 5 Integration of sensors and modules to data acquisition

integration of sensors and modules performed in the third phase of Fig. 3.

All communication between the devices responsible for filter processing, actuator control, and image processing is managed by the central module and performed through an RS-485 bus. In this regard, a communication protocol was developed for data flow control. Table 4 shows the data packet format developed for communication between the modules (FPGA, Raspberry PI, and sensors), which includes a header with 4 fields (i.e., start, address, command, and number of data); a data payload with arbitrary size; a cyclic redundancy check (CRC) to validate the entire packet; and a stop byte. Note that each module is identified with a unique address, allowing the individual control of each actuator. In addition, the modules are connected via the RS-485 standard, making it a serial communication system.

Through the developed communication protocol, the data acquisition sequence can be performed, taking the following steps: (*1*) data from the IMU located at the top of the platform; (*2*) the position data of the markers located at the base of the platform; and (*3*) the position data of the actuators via the encoders. Once the data acquisition is made, the central module sends the pre-processed data to the FPGA to estimate the pose of the Stewart platform.

After each iteration of the quaternion-based EKF, all available data, i.e., data from the IMU, camera, encoders, and the pose estimated by the filter are saved in a data logging. This makes it possible to assess the embedded EKF afterward.

4.4 Validation of Estimated States

To validate this work, we made a comparison between the pose estimated through the quaternion-based Extended Kalman Filter with the pose inferred through the forward kinematics, as illustrated in Fig. 6.

Forward kinematics is an approach that considers the system geometry; however, this technique leads to several solutions. Therefore, it is necessary to model the Stewart platform according to the movement restrictions of each moving part, and then use numerical methods to find the unique solution [17]. In this sense, we use the Nelder-Mead numerical method to minimize a cost function that quantifies the error between the measured value and the estimated value.

To perform the minimization process, we used the `fminsearch` function of the MATLAB software, which minimizes a cost function from an initial value. The initial

Table 4 Data packet format developed for the communication protocol

START	ADDRESS	COMMAND	N DATA	DATA	CRC16	STOP
1 Byte	1 Byte	1 Byte	1 Byte	N Bytes	2 Byte	1 Byte

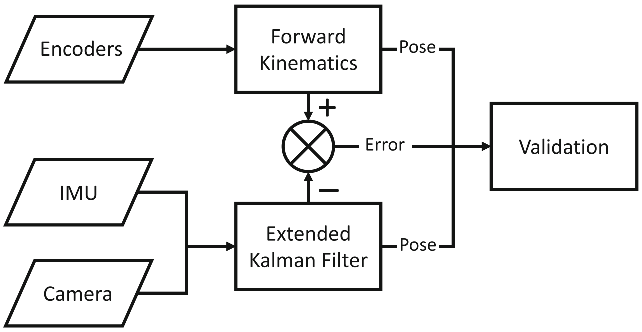


Fig. 6 Validation process of estimated states

position and orientation value can be obtained using the following equations:

$$\begin{aligned}\boldsymbol{B}_{b,i}(\boldsymbol{p}_k, \boldsymbol{q}_k) &= \boldsymbol{B}_i R(\boldsymbol{q}_B) + \boldsymbol{p}_k^T,\\ \boldsymbol{T}_{b,i}(\boldsymbol{p}_k, \boldsymbol{q}_k) &= \boldsymbol{T}_i R(\boldsymbol{q}_T) + \boldsymbol{p}_k^T,\end{aligned} \tag{34}$$

where $R(\boldsymbol{q}_B)$ and $R(\boldsymbol{q}_T)$ are quaternion-based rotation matrices between the global coordinate and the local coordinates of the base and top of the Stewart platform, respectively. The vectors $\boldsymbol{B}_i$ and $\boldsymbol{T}_i$ can be found from (14) and (15). Thus, it is possible to infer the estimated size of actuators $\hat{\boldsymbol{L}}_i$, such as:

$$\hat{\boldsymbol{L}}_i = \sqrt{(\boldsymbol{T}_{b,i} - \boldsymbol{B}_{b,i})^T (\boldsymbol{T}_{b,i} - \boldsymbol{B}_{b,i})}. \tag{35}$$

Therefore, we have:

$$\boldsymbol{\vartheta}_k = \arg\min_{\boldsymbol{\vartheta}_k} J(\boldsymbol{\vartheta}_k) = \arg\min_{\boldsymbol{\vartheta}_k} \sum_{i=1}^{6} (\boldsymbol{L}_i - \hat{\boldsymbol{L}}_i(\boldsymbol{\vartheta}_k))^2, \tag{36}$$

where $\boldsymbol{\vartheta}$ refers to the estimated pose of the platform top, i.e., $\boldsymbol{\vartheta}_k = [\hat{\boldsymbol{p}}_k^T \; \hat{\boldsymbol{q}}_k^T]$. However, because it is a nonlinear optimization problem, in each iteration the algorithm must consider the previous solution $\boldsymbol{\vartheta}_{k-1}$ as the initial value for calculating $\boldsymbol{\vartheta}_k$. With these equations, we calculate the actual pose of the Stewart platform and compare it with the pose estimate of our quaternion-based EKF algorithm.

4.5 Experimental Setup

Table 5 shows the geometric parameters and the rotation matrices associated with the sensors of the Stewart platform used in this work, which is composed of an STMicroelectronics LSM6DS3H IMU and a Logitech C270 camera, both attached to the bottom of the top of the Stewart platform. In addition, to validate our approach, measurements from the encoders mounted in each of the Stewart legs are gathered. The encoders have a resolution of 64 pulses per revolution and are connected to a reduction box with a ratio of 18.75 : 1. To transmit the force, a trapezoidal spindle with a 3 mm pace is used. Based on this, our reference for validation has errors of ±0.0025 mm for each actuator.

Table 5 Stewart platform geometric parameters and sensor rotation matrices

Description	Value
r_B (Bottom radius)	0.35 (m)
r_T (Top radius)	0.25 (m)
φ_B (Angle of each leg to bottom)	$\pi/6$ (rad)
φ_T (Angle of each leg to end-efector)	$\pi/2$ (rad)
R_{IMU} (IMU Rotation Matrix)	$\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
R_{cam} (Camera Rotation Matrix)	$\begin{bmatrix} -0.999 & -0.0116 & -0.003 \\ -0.011 & 0.999 & 0.01 \\ 0.003 & 0.01 & -0.999 \end{bmatrix}$

Figure 7 shows a photo of our experiments, highlighting the four markers with distinct colors located at the base of the Stewart platform. In addition, it is worth mentioning that in our approach the camera position is considered to be the origin of the local coordinate, causing an IMU *offset* regarding the origin that corresponds to $\boldsymbol{O}_{imu} = \left[0.71 \;\; 0.5 \;\; 0.57\right]^T$ meters.

5 Results

This section aims to evaluate the proposed quaternion-based EKF pose estimation algorithm with IMU and visual marker

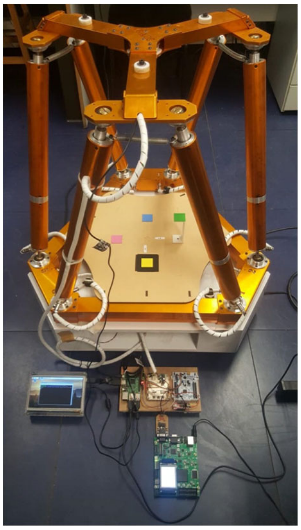


Fig. 7 Photo of the experiments showing the integration of the FPGA and Raspberry PI with the Stewart platform

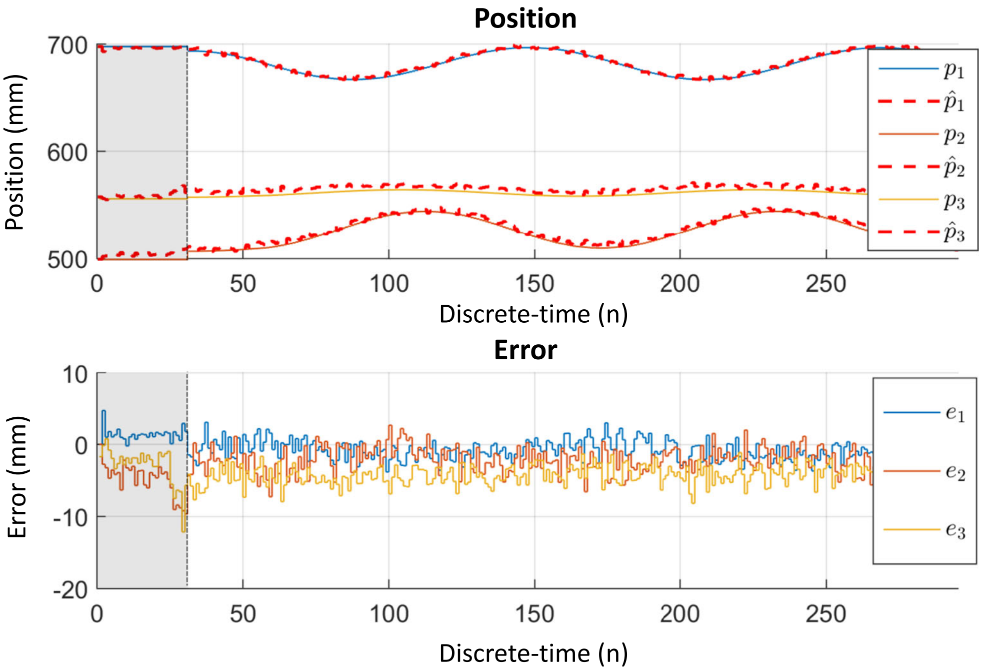


Fig. 8 Position variation and the measurement error of the 1^{st} experiment

data and validate it against the state-of-the-art. Note that our approach is the first to estimate the pose through an FPGA prototype with an ARM Cortex-M3 soft-core. In this sense, the experiments were performed with a low sampling rate (i.e., 0.833 Hz) and, therefore, with a Stewart platform dynamics 100 times slower. This demonstrates the correct implementation of the algorithm and the support to work at low frequency, which will decrease power consumption in a real environment, i.e., in the industry, where an integrated circuit with a hard-core processor is likely to be used.

Three set of experiments were performed to evaluate our quaternion-based EKF algorithm: the first having the markers present all the time, the second forcing the loss of the markers to demonstrate the efficiency of the proposed

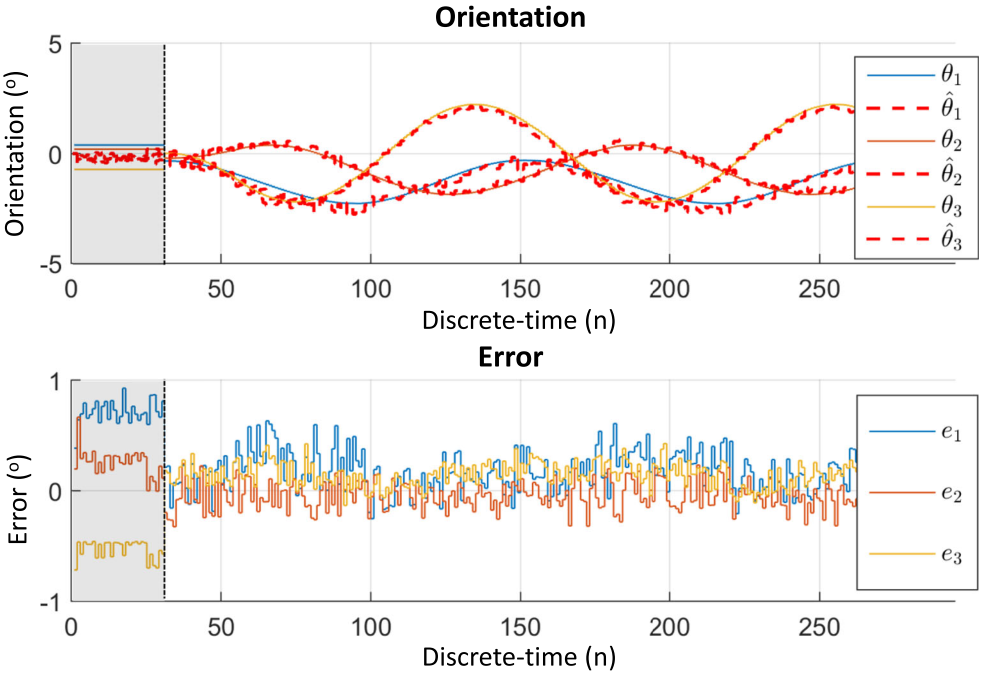


Fig. 9 Orientation variation and the measurement error of the 1^{st} experiment

Table 6 Covariances associated with sensors from the 1^{st} experiment

Sensor	Accelerometer	Gyroscope	Camera
Exp. 1	diag{0.02793, 0.01708, 0.0465}	$10^{-3}\times$diag{0.7401, 0.697, 1.136}	$10^{-3}\times I_8$

filter, and the third lasting twice as long. The results of the experiments were acquired by running the Stewart platform in real-time and saving them to a data log, as shown at the end of the development flow in Fig. 3. Later, the results are validated with the pose inferred through the forward kinematics, as shown in Section 4.4.

The first experiment is dedicated to assessing the pose estimation error of our quaternion-based EKF algorithm. The top part of Fig. 8 shows the position variation of the Stewart platform during the experiment, where the continuous line represents the actual position and the dotted line the pose estimated by the quaternion-based EKF algorithm. To make the graph more didactic, the bottom part of Fig. 8 highlights the position error. Complementary, Fig. 9 shows the orientation of the top of the Stewart platform, where the results are presented through the Euler angles (*roll*, *pitch* and *yaw*).

For each application startup, a sensor data package is acquired with the Stewart platform at rest and then the covariance matrix of the sensors Q_z and their respective bias are calculated, as shown in Table 6. This application startup is represented by the gray shaded part in Figs. 8 and 9 from $n = 0$ to $n = 30$, and we will not consider the error presented at those moments, since it serves as an automatic calibration. On the other hand, the covariance of the model was previously defined empirically and consists of a matrix given by $Q_x = \begin{bmatrix} 10^{-5}\times I_{19\times 19} & 0_{19\times 6} \\ 0_{6\times 19} & 0_{6\times 6} \end{bmatrix}$.

Figures 8 and 9 show that the trajectories estimated by the filter behave as expected in relation to the real states.

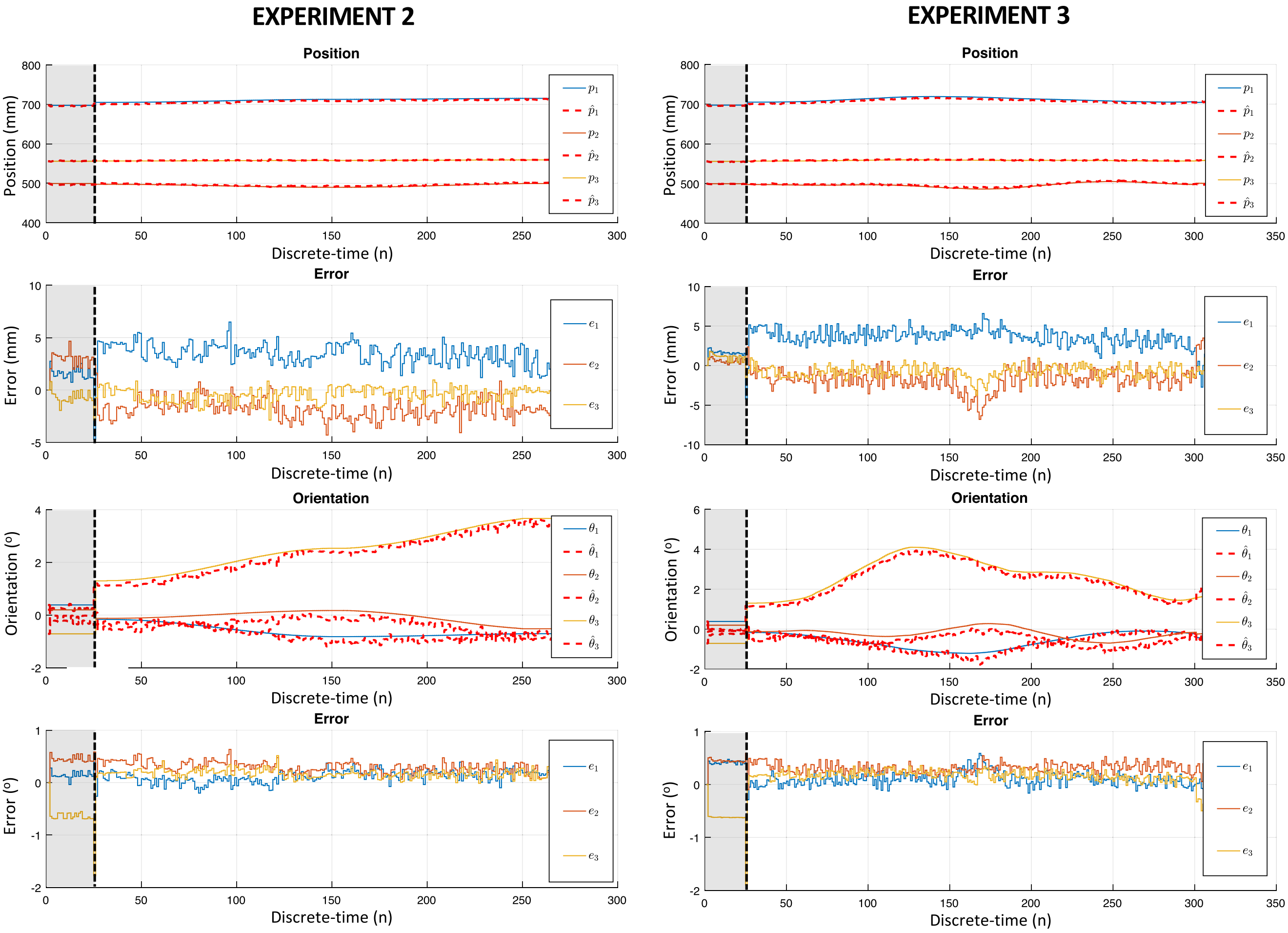


Fig. 10 Two additional experiments varying the position and orientation of the Stewart platform to determine the error found by the proposed pose estimate under normal conditions

Table 7 Statistical analysis of three experiments performed under normal conditions

	Position Mismatch (mm)				Euler angles error (º)		
Description	Exp. 1	Exp. 2	Exp. 3	Description	Exp. 1	Exp. 2	Exp. 3
mean x: p_1	-0.7119	3.3123	3.4536	mean roll: θ_1	0.1630	0.1238	0.0847
mean y: p_2	-2.0394	-1.8161	-1.4119	mean pitch: θ_2	-0.0452	0.2790	0.3017
mean z: p_3	-4.1676	-0.4142	-0.6918	mean yaw: θ_3	0.1456	0.1712	0.1669
σ_{p_1}	1.3512	1.0078	1.1886	σ_{θ_1}	0.1792	0.1064	0.1322
σ_{p_2}	1.7648	0.8788	1.6779	σ_{θ_2}	0.1289	0.1100	0.0945
σ_{p_3}	1.3701	0.6304	0.8092	σ_{θ_3}	0.1054	0.0807	0.1042
$\max(\lvert p_1\rvert)$	3.7803	6.4879	6.5842	$\max(\lvert\theta_1\rvert)$	0.6304	0.3912	0.5879
$\max(\lvert p_2\rvert)$	7.2305	0.8578	4.2297	$\max(\lvert\theta_2\rvert)$	0.3254	0.6328	0.5442
$\max(\lvert p_3\rvert)$	8.1471	0.9618	1.4290	$\max(\lvert\theta_3\rvert)$	0.4268	0.5138	0.3584

In addition to these results, Fig. 10 shows two other similar experiments varying the position and orientation of the Stewart platform. This helps to determine the maximum error found by the proposed pose estimate under normal conditions. Table 7 presents the mean, standard deviation (σ) and the highest absolute value of the estimation errors for each orientation and position axis from the FPGA acquired data. This shows that the developed filter was able to estimate the pose with an error below 8.14 mm and 0.63º error, as highlighted in red in Table 7. Note that the impact of filter accuracy will depend on the application used. For example, the Stewart platform can be used for both flight simulators [18] and precision surgery [19]. While in the former case this precision does not have much impact, it is remarkable that in a precision surgery, the accuracy of the filter can be decisive for saving or not saving a life, showing why filter optimization is so important.

Considering the literature (summary shown in Table 1), we compared our results with other methods to obtain the pose estimate without using the forward kinematics approach, in addition to validate and highlight the results obtained. Zuo et al. [34] presented a vision-based approach to estimate the pose of a Stewart platform and obtained position errors around 3.00 mm, 0.88 mm and 1.09 mm, for the x, y and z-axis and orientation errors of 0.079º, 0.316º and 0.288º, for roll, pitch and yaw, respectively. However, to estimate the pose in real-time, their stereo vision approach use at least two cameras, requiring a higher processing power compared to our approach. In addition, the charge-coupled device (CCD) cameras used are large and power-consuming, which is no longer recommended for use in industry. Another vision-based work is presented by Yoon et al. [35], their Kalman-Filter-based visual tracking algorithm presents a worst-case position error of 9.2 mm and an orientation error of 0.67º. This proves that our quaternion-based EKF algorithm is more accurate than Yoon et al. [35] (i.e., our worst-case is 8.14 mm and 0.63º), in addition to supporting online applications, while they rely on pre-recorded scenes.

Similar to our approach, but visioning other applications, i.e., not applied to a Stewart platform, Araguás et al. [36] presented results that only estimates the orientation of an UAV with an error around 0.22º. Also, Du and Zhang [40] presented the pose estimate of a serial manipulator with a maximum error of 1.99 mm in position and 1.02º in orientation. However, their initial serial manipulator error is 9.01 mm, while our Stewart platform (i.e., a parallel manipulator) could reach hundreds of millimeters. Lastly, Mirzaei and Roumeliotis [39] presented estimation errors around 8.0 mm in position and 0.1º in orientation, where the initial uncertainty was 90 mm and 6º, respectively. These comparisons reinforce the prominence of our approach using a low-cost IMU sensor (LSM6DS3H) and a conventional webcam (Logitech C270) with simple geometry markers to obtain excellent pose estimates from the Stewart platform, with a maximum error of 8.14 mm in position and 0.63º in orientation, which proves the quality of the developed quaternion-based EKF algorithm.

The second set of experiments aim to assess the reliability of our algorithm when dealing with the loss of camera markers and also with the loss of all input data. When it happens, the implemented filter algorithm reuses the data from the previous instant. Figure 11 shows the filter performance under such conditions.

Similar to the first set of experiments, in Fig. 11 the gray shaded part means the application startup. As each sample of the discrete-time is equivalent to 1.2 seconds, the Stewart platform is at rest for 24 seconds (i.e., $n = 20$). In addition, the covariance associated with the sensors in this experiment are shown in Table 8. On the other hand, the differences are

Fig. 11 Assessment to determine pose estimation mismatches when dealing with the loss of input data

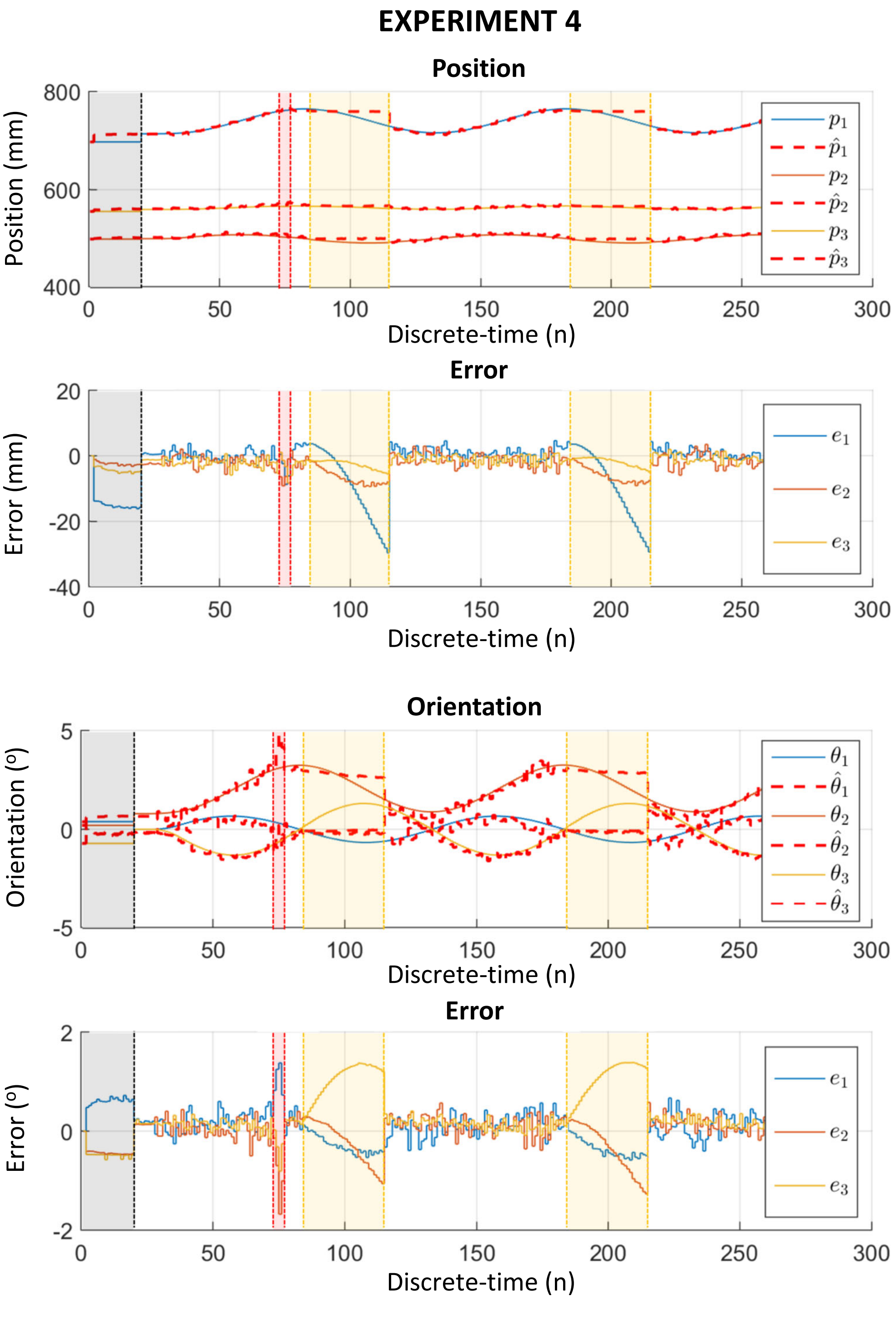


Table 8 Covariances associated with sensors from the 4^{th} experiment

Sensor	Accelerometer	Gyroscope	Camera
Exp. 4	diag{0.01158, 0.00806, 0.01276}	$10^{-3}\times$diag{0.9007, 1.2848, 0.3798}	$10^{-3}\times I_8$

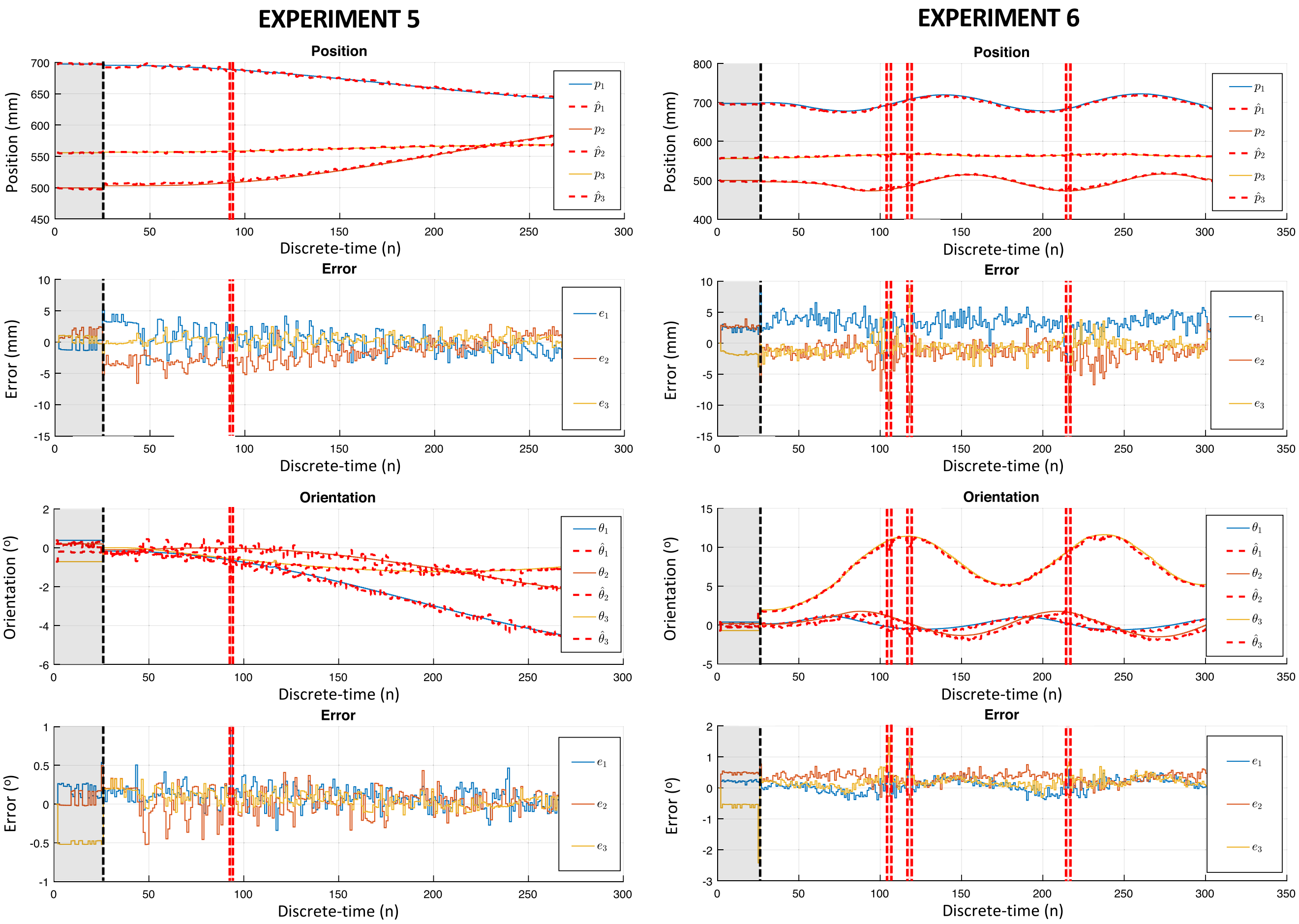


Fig. 12 Two additional experiments varying the position and orientation with data loss to show the quick recovery time of the filter to converge to the correct estimated pose values

in the conditions of the experiments. In Fig. 11, we force the loss of markers in two different moments, when $n = 85$ and $n = 185$, as shown in the orange shaded parts. In addition, we also force the loss of all input data when $n = 73$, as shown in the red shaded part. As a consequence of this bias of information, the filter deviates from the real data acquired from the platform. This is because the algorithm keeps performing the update step with the previous data measurements. However, once the markers are back in the camera's field of view in the orange shaded parts (i.e., $n = 115$ and $n = 215$) or all input data is back in the red shaded part (i.e., $n = 77$), the filter converges to the correct pose estimate values in only one sample. To support this analysis, Fig. 12 shows two more experiments with the same objective. It demonstrates that (*1*) the algorithm is dependent on all input data and (*2*) the algorithm is able to quickly recover and converge to the optimal pose estimation value. Also note that the design choice to continue using the previous marker values has increased the reliability of the developed filter, as the error takes a little longer to grow.

Finally, the last experiment aims to analyze the behavior of the proposed quaternion-based EKF pose estimation with IMU and visual marker data over time using sinusoidal trajectories. The experiment is evaluated under normal conditions, where the Stewart platform legs constantly change direction. The results shown in Fig. 13 suggest that the accumulation of data acquired over time slightly improves the pose estimate of the Stewart platform. Therefore, the mismatches are related to the initial estimates, as assessed in the first set of experiments; and mainly, the loss of input data, as assessed in the second set of experiments.

6 Conclusions

Nowadays, more and more six degrees of freedom systems are present in industrial automation, which makes their efficiency and diversity of use a constant challenge. In this work, we proposed a quaternion-based Extended Kalman Filter pose estimation algorithm with IMU and visual

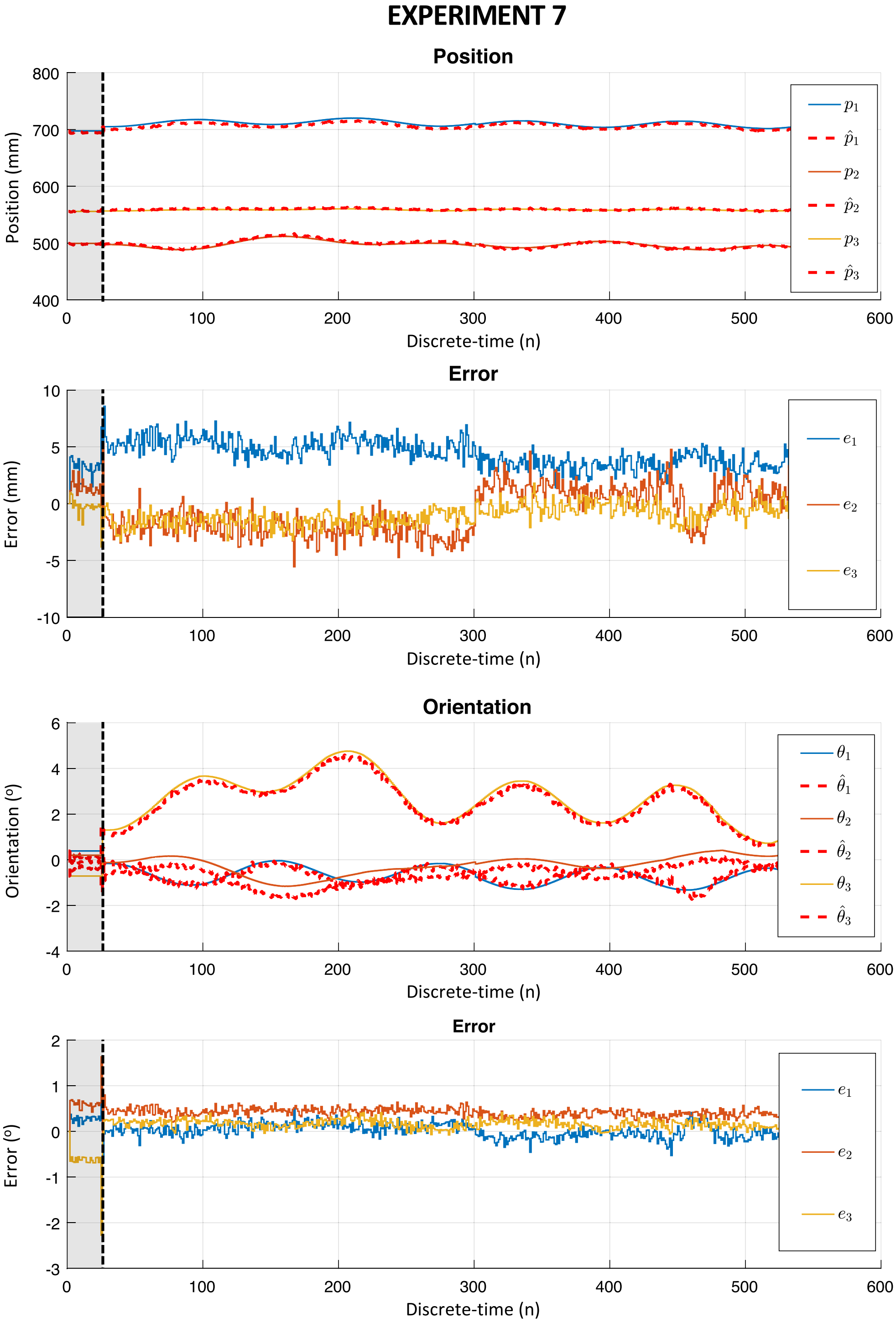


Fig. 13 An experiment conducted with longer exposure time to analyze the behavior of the proposed quaternion-based EKF pose estimation

marker data, which stands out for its low computational effort and greater reliability to deal with partial sensors availability. In addition, the position and orientation of 6–DOF systems are estimated in real-time. To validate and evaluate our proposed pose estimation algorithm, the implementation on a Stewart platform has been carried out. Furthermore, our pose estimation solution was the first to be embedded on an FPGA device with commercial processors, demonstrating the ease of reuse of this approach by other works. Further research work intend to decouple the algorithm to be used in multicore systems, this will bring performance gains and freedom to be implemented in different ways.

Author Contributions All authors contributed to the study and development of this work, with the following responsible for the sections: Introduction: Rodrigo Medeiros and Rafael Garibotti; Literature Review: all authors; Background: Rodrigo Medeiros and Guilherme Pimentel; Algorithm's implementation: Rodrigo Medeiros; Development flow: all authors; Experimental setup and data collection: Rodrigo Medeiros; Analysis and discussion of results: all authors; Writing - original draft preparation: Rodrigo Medeiros and Rafael Garibotti; Writing - review and editing: all authors; Supervision: Rafael Garibotti and Guilherme Pimentel.

Funding This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Availability of Data and Materials The source code and raw data used in this work are available at https://github.com/rafaelgaribotti/JINT-D-20-00225.

Declarations

Conflict of Interests The authors declare that in addition to funding, there is no conflict of interest or competing interests.

References

1. Mattila, J., Koivumäki, J., Caldwell, D.G., Semini, C.: A survey on control of hydraulic robotic manipulators with projection to future trends. IEEE/ASME Trans. Mechatron. **22**(2), 669–680 (2017)
2. Zhang, S., Dong, Y., Ouyang, Y., Yin, Z., Peng, K.: Adaptive neural control for robotic manipulators with output constraints and uncertainties. IEEE Trans. Neural Netw. Learn. Syst. **29**(11), 5554–5564 (2018)
3. Markus, E.D., Agee, J.T., Jimoh, A.A.: Flat control of industrial robotic manipulators. Robot. Auton. Syst. **87**(C), 226–236 (2017)
4. Sharma, U., Rawal, V., Tayal, V.: An overview on multiple unmanned aerial vehicle control through single controller. Int. J. Comput. Appl. **180**(49), 8–12 (2018)
5. Aboudonia, A., El-Badawy, A., Rashad, R.: Active anti-disturbance control of a quadrotor unmanned aerial vehicle using the command-filtering backstepping approach. Nonlinear Dynam. **90**(1), 581–597 (2017)
6. Xiang, X., Yu, C., Zhang, Q.: Robust fuzzy 3D path following for autonomous underwater vehicle subject to uncertainties. Comput. Oper. Res. **84**(C), 165–177 (2017)
7. Cai, K., Tian, Y., Liu, X., Fatikow, S., Wang, F., Cui, L., Zhang, D., Shirinzadeh, B.: Modeling and controller design of a 6–DOF precision positioning system. Mech. Syst. Signal Process. **104**, 536–555 (2018)
8. Liu, M., Zhou, H., Pang, A.: Research on motion control system of 6–DOF robotic arm. In: International Conference on Modelling, Identification and Control (ICMIC), pp. 53–61 (2019)
9. Najm, A.A., Ibraheem, I.K.: Nonlinear PID controller design for a 6–DOF UAV quadrotor system. Eng. Sci. Technol. Int. J. **22**(4), 1087–1097 (2019)
10. Li, J., Bernardos, A.M., Tarrío, P., Casar, J.R.: A combined vision-inertial fusion approach for 6–DOF object pose estimation. In: International Conference on Machine Vision (ICMV), vol. 9445, pp. 238–249 (2015)
11. Xu, D., Li, Y.F.: A new pose estimation method based on inertial and visual sensors for autonomous robots. In: IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 405–410 (2007)
12. Ma, T., Song, Z., Xiang, Z., Dai, J.S.: Trajectory tracking control for flexible-joint robot based on extended Kalman filter and PD control. Front. Neurorobot. **13**(25), 1–10 (2019)
13. Dong, G., Zhu, Z.H.: Autonomous robotic capture of non-cooperative target by adaptive extended Kalman filter based visual servo. Acta Astronaut. **122**, 209–218 (2016)
14. Chung, J.H., Pak, J.M., Ahn, C.K., You, S.H., Lim, M.T., Song, M.K.: Particle filtering approach to membership function adjustment in fuzzy logic systems. Neurocomputing **237**(10), 166–174 (2017)
15. Diebel, J.: Representing attitude: Euler angles, unit quaternions, and rotation vectors. Matrix **58**(15), 1–35 (2006)
16. Debruin, J.: Control systems for mobile satcom antennas. IEEE Control. Syst. Mag. **28**(1), 86–101 (2008)
17. Cardona, M.: A new approach for the forward kinematics of general stewart-gough platforms. In: Proceedings of the 2015 IEEE Thirty Fifth Central American and Panama Convention (CONCAPAN XXXv) (2015)
18. Stewart, D.: A platform with six degrees of freedom. Inst. Mechan. Eng. **180**(1), 371–386 (1965)
19. Kratchman, L.B., Blachon, G.S., Withrow, T.J., Balachandran, R., Labadie, R.F., Webster, R.J.: Design of a bone-attached parallel robot for percutaneous cochlear implantation. IEEE Trans. Biomed. Eng. **58**(10), 2904–2910 (2011)
20. Ho, P.T.P., Altamirano, P., Chang, C.H., Chang, S.H., Chang, S.W., Chen, C.C., Chen, K.J., Chen, M.T., Han, C.C., Ho, W.M., et al: The Yuan-Tseh Lee array for microwave background anisotropy. Astrophys. J. **694**(2), 1610–1618 (2009)
21. Mello, C.B.: Controle De Trajetória De Uma Plataforma Stewart Para Simulacão De Transferência De Carga Fora De Porto. Master's thesis, Federal University of Rio de Janeiro (2011)
22. de Sousa, R.A., Ferreira, J., de Farias, J.S., Torrão, J., Afonso, D., Martins, M.: SPIF-A: On the development of a new concept of incremental forming machine. Struct. Eng. Mech. **49**(5), 645–660 (2014)
23. Garibotti, R., Butko, A., Ost, L., Gamatié, A., Sassatelli, G., Adeniyi-Jones, C.: Efficient embedded software migration towards clusterized distributed-memory architectures. IEEE Trans. Comput. **65**(8), 2645–2651 (2016)
24. Luces, M., Mills, J.K., Benhabib, B.: A review of redundant parallel kinematic mechanisms. J. Intell. Robot. Syst. **86**, 75–198 (2017)
25. Navvabi, H., Markazi, A.H.D.: New afsmc method for nonlinear system with state-dependent uncertainty: Application to hexapod robot position control. J. Intell. Robot. Syst. **95**, 61–75 (2019)
26. M'hiri, S.A., Romdhane, N.M.B., Damak, T.: New forward kinematic model of parallel robot par4. J. Intell. Robot. Syst. **96**, 283–295 (2019)
27. Liu, K., Lewis, F., Lebret, G., Taylor, D.: The singularities and dynamics of a stewart platform manipulator. J. Intell. Robot. Syst. **8**(3), 287–308 (1993)
28. Dietmaier, P.: The stewart gough platform of general geometry can have 40 real postures. In: Proceedings of ARK, Strobl, Austria, June 29- July 4, pp. 7–16 (1998)
29. Coronado, E., Maya, M., Cardenas, A., Guarneros, O., Piovesan, D.: Vision-based control of a delta parallel robot via linear camera-space manipulation. J. Intell. Robot. Syst. **85**, 93–106 (2017)
30. Song, S., Kwon, D.: New direct kinematic formulation of 6 D.O.F stewart-gough platforms using the tetrahedron approach. Trans. Control Autom. Syst. Eng. **4**(3), 217–223 (2002)
31. Kumar, P.R., Behera, A.K., Bandyopadhyay, B.: Robust finite-time tracking of stewart platform: a super-twisting like observer-based forward kinematics solution. IEEE Trans. Ind. Electron. **64**(5), 3776–3785 (2017)
32. Yang, C., He, J., Han, J., Liu, X.: Real-time state estimation for spatial six-degree-of-freedom linearly actuated parallel robots. Mechatronics **19**(6), 1026–1033 (2009)

33. Rendon-Mancha, J., Cardenas, A., Garcia, M., Gonzalez-Galvan, E., Lara, B.: Robot positioning usingcamera-space manipulation with a linear camera model. IEEE Trans. Robot. **26**(4), 726–733 (2010)
34. Zuo, A., Wu, Q.M.J., Gruver, W.A.: Stereo vision guided control of a stewart platform. In: Proceedings of the IEEE Internatinal Symposium on Intelligent Control, pp. 125–130 (2002)
35. Yoon, Y., Kosaka, A., Kak, A.C.: A new kalman-filter-based framework for fast and accurate visual tracking of rigid objects. IEEE Trans. Robot. **24**(5), 1238–1251 (2008)
36. Araguás, G., Paz, C., Gaydou, D., Paina, G.P.: Quaternion-based orientation estimation fusing a camera and inertial sensors for a hovering UAV. J. Intell. Robot. Syst. **77**, 37–53 (2015)
37. Erdem, A.T., Ercan, A.O.: Fusing inertial sensor data in an extended kalman filter for 3d camera tracking. IEEE Trans. Image Process. **24**(2), 538–548 (2015)
38. Nützi, G., Weiss, S., Scaramuzza, D., Siegwart, R.: Fusion of IMU and vision for absolute scale estimation in monocular. SLAM J. Intell. Robot. Syst. **61**(1-4), 287–299 (2011)
39. Mirzaei, F.M., Roumeliotis, S.I.: A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation. IEEE Trans. Robot. **24**(5), 1143–1156 (2008)
40. Du, G., Zhang, P.: Online serial manipulator calibration based on multisensory process via extended kalman and particle filters. IEEE Trans. Ind. Electron. **61**(12), 6852–6859 (2014)
41. Miletović, I., Pool, D., Stroosma, O., van Paassen, M., Chu, Q.: Improved stewart platform state estimation using inertial and actuator position measurements. Control. Eng. Pract. **62**, 102–115 (2017)
42. Alaimo, A., Artale, V., Milazzo, C., Ricciardello, A.: Comparison between euler and quaternion parametrization in uav dynamics. In: 11th International Conference of Numerical Analysis and Applied Mathematics 2013: ICNAAM 2013. AIP Conference Proceedings (2013)
43. Lee, C., Salcic, Z.: A fully-hardware-type maximum-parallel architecture for Kalman tracking filter in FPGAs. In: International Conference on Information, Communications and Signal Processing (ICICS), vol. 2, pp. 1243–1247 (1997)
44. Bonato, V., Peron, R., Wolf, D.F., de Holanda, J.A.M., Marques, E., Cardoso, J.M.P.: An FPGA implementation for a kalman filter with application to mobile robotics. In: International Symposium on Industrial Embedded Systems (SIES), pp. 148–155 (2007)
45. Guo, H., Chen, H., Xu, F., Wang, F., Lu, G.: Implementation of EKF for vehicle velocities estimation on FPGA. IEEE Trans. Ind. Electron. **60**(9), 3823–3835 (2012)
46. NXP: LPC1800 Series MCUs. Available at https://www.nxp.com/docs/en/brochure/75017637.pdf (2015)
47. TI: OMAP4 mobile applications platform. Available at https://www.ti.com/lit/ml/swpt034b/swpt034b.pdf (2011)
48. Garibotti, R., Reagen, B., Shao, Y.S., Wei, G.Y., Brooks, D.: Using dynamic dependence analysis to improve the quality of high-level synthesis designs. In: IEEE International Symposium on Circuits and Systems (ISCAS), p. 1–4 (2017)
49. Olenick, R.P., Apostol, T.M., Goodstein, D.L.: The Mechanical Universe: Introduction to Mechanics and Heat. Cambridge University Press, Cambridge (2008)
50. Vasquez, X.M.Z.: Posicionamento De Múltiplos Objetos a Partir De Visão Estéreo. Master's thesis, Pontifical Catholic University of Rio Grande do Sul (2015)
51. Welch, G., Bishop, G.: An introduction to the Kalman filter. In: ACM SIGGRAPH, pp. 19–34 (2001)
52. Dasgupta, B., Mruthyunjaya, T.: The stewart platform manipulator: A review. Mechan. Mach. Theor. **35**(1), 15–40 (2000)
53. Kucuk, S.: Serial and Parallel Robot Manipulators: Kinematics. Dynamics, Control and Optimization. IntechOpen (2012)
54. de Faria, P.F.S.R., Castro, R.S., Salton, A.T., Flores, J.V.: Quaternion-based dynamic control of a 6–DOF stewart platform for periodic disturbance rejection. In: IEEE Conference on Control Applications (CCA), pp. 1191–1196 (2016)
55. Markley, F.L., Crassidis, J.L.: Fundamentals of spacecraft attitude determination and control, vol. 33. Springer, New York (2014)
56. Marins, J.L., Yun, X.iaoping., Bachmann, E.R., McGhee, R.B., Zyda, M.J.: An extended Kalman filter for quaternion-based orientation estimation using MARG sensors. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), vol. 4, pp. 2003-2011 (2001)
57. Garibotti, R., Reagen, B., Shao, Y.S., Wei, G.Y., Brooks, D.: Assisting high-level synthesis improve spmv benchmark through dynamic dependence analysis. IEEE Trans. Circ. Systems II Express Briefs **65**(10), 1440–1444 (2018)
58. Garibotti, R., Ost, L., Butko, A., Reis, R., Gamatié, A., Sassatelli, G.: Exploiting memory allocations in clusterised many-core architectures. IET Comput. Digit. Techniq. **13**(4), 302–311 (2019)
59. Teague, M.R.: Image analysis via the general theory of moments. J. Opt. Soc. Am. **70**(8), 920–930 (1980)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Rodrigo Alves Medeiros was born in Cuiabá, Brazil, in 1993. He received his Control and Automation Engineering degree and Master degree in Electrical Engineering from the Pontifical Catholic University of Rio Grande do Sul (Brazil) in 2018 and 2020, respectively. His research interests include signal processing and control of (non)linear systems.

Guilherme Araujo Pimentel received his Control and Automation Engineering degree and Master degree in Electrical Engineering from the Pontifical Catholic University of Rio Grande do Sul (Brazil) in 2007 and 2010, respectively. He obtained a joint PhD degree in Science of Engineering - University of Mons (Belgium) - and Mathematics & Modeling - University of Montpellier (France) - in 2015. Between 2015 and 2017, he had a postdoctoral position in the Department of Automation and Energy of Universidade Federal do Rio Grande do Sul, in Porto Alegre, Brazil. Between August 2015 and January 2021, he was an Associated Professor at School of Technology from Pontifical Catholic University of Rio Grande do Sul (PUCRS). Since February 2021, he has a research position in the SECO Department (UMONS). His main research interests are in system identification, modeling and control of (non)linear systems.

Rafael Garibotti is currently an Associate Professor at PUCRS University. Former Visiting Scholar at Université Grenoble Alpes, France. Former Postdoctoral Fellow at both the prestigious School of Engineering and Applied Sciences of Harvard University and UFRGS, Brazil. He received his PhD and MSc. Degree in Microelectronics, respectively from the University of Montpellier and EMSE, France and his BSc. Degree in Computer Engineering from PUCRS University, Brazil. His research activity focuses on AI safety, robotics and autonomous systems, multicore architectures, hardware accelerator and robust deep learning.