# Cooperative Position and Orientation Estimation for Multi-Vehicle Systems

**Francisco José Machado e Moura**

Instituto Superior Técnico
Universidade de Lisboa, Portugal
Email: franciscojmmoura@tecnico.ulisboa.pt

*Cooperative estimation and control are thriving topics of research within the field of multi-agent systems. Cooperation may enable or improve performance in task execution, through local sensing and exchange of information.*

*With local mapping being undertaking complex and with high energy consumption. In a GPS denied environment without previous knowledge of the structure, autopilot tasks cannot be executed, making cooperative estimation a better suited option. This thesis is focused on investigating different interaction topologies and the adequate selection of local measurements to enable the collective estimation of position and orientation of all vehicles.*

*This thesis provides an overall view on the fundamental concepts of an UAV's position and heading estimation algorithm for a very specific set of conditions, such as, the lack of GPS signal under a bridge deck.*

*Filters are fundamental to obtain the best estimation possible with the sensors and models available. Due to its optimal estimation capabilities, the Kalman Filter, and its different variants have been used in autonomous navigation for several years. Therefore, 3 of the used filtering algorithms are Kalman Filter based.*

## 1 Introduction

In recent years, unmanned aerial vehicles, also known as UAVs, have been capturing the attention and interest in a broad range of applications, such as search and coverage missions, exploration, cooperative manipulation and control. In each of these cases, the problem of cooperative localisation in a group of drones has to be solved. Cooperative localisation consists in improving the positioning capacity of each drone through the exchanges of information with other drones in the group [1].

Relative localisation between UAVs is a requisite for cooperative localisation and orientation estimation in the case when absolute localisation information such as information from global positioning system (GPS) is unavailable or to inaccurate to be used, which can happen indoors, in urban environments and even forest and remote locations. Taking this in consideration, relative localisation is a key parameter in cooperative UAV systems [2].

The problem of cooperative position and orientation estimation in multi-agent systems is formulated in this thesis considering a flight in formation with three drones labelled $1, 2, 3$. The formation is composed by a leader aircraft, 3, at unknown location at time $t$, followed by two drones, 1 and 2, each equipped with a GPS receiver.

A convenient solution for relative location is to obtain the distance and bearing information using cameras and AI methods. Nevertheless,there is the limitation of cameras only being able of operate within a limited range and are prone to suffer from occlusion and lighting conditions. In opposition, distance measurements can be obtained using different sensors such as ultra-wideband (UWB), radars, and lidars, which can operate over a much larger range [2].

There are, already, several methods used for solving cooperative localisation problems, such as the Extended Kalman Filter (EKF) for a centralised system, or, if the computation is decentralised and the communication is unreliable, other techniques like Covariance Intersection or Interleaved Update. Approaches that assume bounded errors using polytopes and linear programming algorithms have also been proposed [1].

The work is focused on investigating the correlation between number of drones used and the results in the cooperative estimation of the position. The different interaction topologies between drones and the adequate selection of local measurements. As well as the better suited filtering techniques to enable the collective estimation of position and orientation of all vehicles taking in consideration the problem constraints.

## 2 Background

The first concept introduced is orientation. In order to understand orientation, first of all, two concepts must be introduced, the local and inertial referential. These two concepts are fundamental to the use of Euler angles, the heading parameters, and for a proper sensor's output usage. In this

subsection, the mathematics behind Euler angles and the rotation matrix are key to the understanding of the heading's representation. And, therefore, major for a suitable description of orientation sensors. The sensors used are the rate gyroscope, the accelerometer and the magnetometer.

Following the orientation overview, the positioning sensors must be described as well. Once the correct estimation of the third drone position lies on it. The sensors used are the GPS and two vision based algorithms whose output is the relative position and relative orientation between drones.

The last concept mentioned in this section is filtering. Filters are fundamental to obtain the best estimation possible with the sensors and models available. The majority of the filters presented are variations of the Kalman Filter. These variations can be due to the non-linear nature of the model, Extended Kalman Filter, or a Kalman Filter interpretation of Complementary Filtering.

## 2.1 Orientation

### 2.1.1 Local and Inertial Referential

Each vehicle has its own local referential, $\{V_i\}$, $i \in [1,2,3]$, sympathetic with the drone's kinematic, that is defined by its position, ${}^{I}\vec{p}_i$, $i \in [1,2,3]$, and orientation, $[\phi_i, \theta_i, \psi_i]$, $i \in [1,2,3]$, in an inertial referential, $\{I\}$.

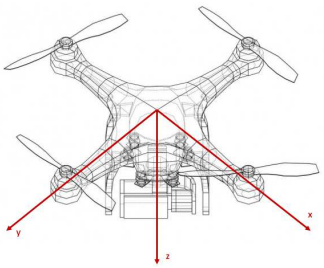

Fig. 1. Local Referential.

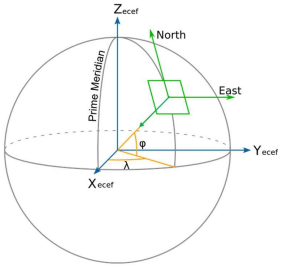


Fig. 2. Inertial Referential.

Each aircraft has its position expressed in the inertial referential as:

$$ {}^{I}\vec{p}_i = (x_i, y_i, z_i) \tag{1} $$

where, $(x_i, y_i, z_i)$, $i \in [1,2,3]$, represents the location of the ith aircraft in a coordinate system external to all vehicles.

### 2.1.2 Euler Angles.

Intuitively the orientation is defined by three angles, the Euler angles, known as roll, pitch and yaw.

Roll is angle of rotation related to the x-axis, whose notation is $\phi$. Pitch is the angle of rotation related to the y-axis, whose notation is $\theta$. Yaw is the angle of rotation related to the z-axis, whose notation is $\psi$.

This representation of Euler angles are shown in Figure 3.

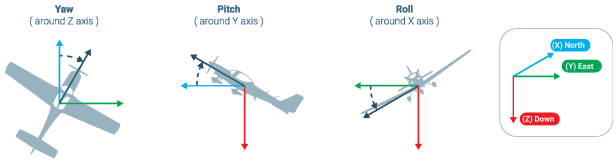


Fig. 3. Euler angles: sequence of standard rotations.

The similarity between linear coordinates and angular coordinates makes Euler angles very intuitive, but unfortunately they have a huge disadvantage, the possible loss of a degree of freedom (Gimbal Lock). A potential solution to the gimbal lock is to represent the orientation in other way that not with Euler angles. Alternative representations for orientation that can be adopted are rotation matrices, quaternions, or direct cosine matrices (DCM).

## 2.2 Rotation Matrices

In order to avoid singularities, rotation matrices are used in this thesis instead of Euler angles [9]. A vector defined at any local referential, $\{V_i\}$, can be transformed into the inertial referential, $\{I\}$, employing sequentially the three rotations, each one given by its rotation matrix.

The Roll rotation matrix of the $ith$ aircraft is defined as:

$$ {}^{I}_{i}R(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix} \tag{2} $$

The Pitch rotation matrix of the $ith$ aircraft is defined as:

$$ {}^{I}_{i}R(\theta) = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{3} $$

The Yaw rotation matrix of the $ith$ aircraft is defined as:

$$ {}^{I}_{i}R(\psi) = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4} $$

Therefore, the Rotation Matrix of the $ith$ aircraft, ${}^{I}_{i}R(\psi){}^{I}_{i}R(\theta){}^{I}_{i}R(\phi)$, with notation $c_\xi = cos(\xi)$ and $s_\xi = sin(\xi)$ is defined as:

$$ {}^{I}_{i}R = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\theta & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \tag{5} $$

## 2.3 Rate Gyroscope

A rate gyro is a type of gyroscope that indicates angular rates, that is, $\omega = [p,q,r]$ without a fixed point of reference.

Although there is no direct correspondence between the angular rates $[p,q,r]$ and the derivatives of Euler angles, the following approximation for small roll and pitch angles can be used: $[p,q] \approx [\dot{\phi},\dot{\theta}]$.

Rate gyros are defined by a drift term and zero mean Gaussian noise, resulting from the manufacturing process.

$$\begin{cases} y_{gyro,x} = p + \beta_{gyro,z} + \eta_{gyro,x}, \eta_{gyro,x} \sim N(0, 0.0169) \\ y_{gyro,y} = q + \beta_{gyro,z} + \eta_{gyro,y}, \eta_{gyro,y} \sim N(0, 0.0169) \\ y_{gyro,z} = r + \beta_{gyro,z} + \eta_{gyro,z}, \eta_{gyro,z} \sim N(0, 0.0169) \end{cases} \quad (6)$$

Where, $[p,q,r]$, represents the angular rates, $\beta_{gyro}$ the Bias and $\eta_{gyro}$ the sensor's noise in the readings. The gyro noise, $\eta_{gyro}$, is driven by noise of ARW (Angular Random Walk), and a Gaussian can be defined as being zero mean, $\mu \approx 0$, $\sigma_{gyro} = 0.0169 s^{-1}$ and $\eta_{gyro} \sim N(0, 0.0169)$ [10].

The bias is a drift term driven by noise of RRW (Rate Random Walk) and is defined as [3]:

$$f(t) = \begin{cases} \beta_{gyro,x} = -0.0113 \\ \beta_{gyro,y} = 0.0322 \\ \beta_{gyro,z} = 0.0115 \end{cases} \quad (7)$$

### 2.4 Accelerometer

An accelerometer is a sensor that measures proper acceleration. Proper acceleration is the acceleration of a body in its own instantaneous rest frame. This is different from coordinate acceleration, which is acceleration in a fixed coordinate system.

The accelerometer can be modelled as the following [4]:

$$\begin{cases} y_{acc,x} = g sen(\theta) + \eta_{acc,x} \\ y_{acc,y} = -g cos(\theta) sen(\phi) + \eta_{acc,y} \\ y_{acc,z} = -g cos(\theta) cos(\phi) + \eta_{acc,z} \end{cases} \quad (8)$$

The noise of the accelerometer:

$$\begin{cases} \eta_{acc,x} \sim N(0, 0.005112) \\ \eta_{acc,y} \sim N(0, 0.007291) \\ \eta_{acc,z} \sim N(0, 0.1614) \end{cases} \quad (9)$$

This sensor is used to obtain measurements of $[\phi,\theta]$, thus an extra sensor to obtain the measurement of $\psi$ must be incorporated.

### 2.5 Magnetometer

The magnetic field sensors can be used as a compass and therefore determine the orientation of the sensor relative to the magnetic north pole. The output is, intuitively, the yaw value, $\psi$, plus white noise, $\eta_{mag}$ [5].

$$y_{mag} = \psi + \eta_{mag}, \qquad \eta_{mag} \sim N(0, 12.399) \quad (10)$$

## 3 Position

The position estimates can be obtained using three sets of measurements: GPS coordinates of two aircraft, inter-vehicle distance and inter-vehicle angle.

Each aircraft has its position regarding the inertial referential, $\{I\}$, defined as:

$${}^{I}\vec{p}_i = (x_i, y_i, z_i), i \in [1,2,3] \quad (11)$$

### 3.1 GPS

GPS, or Global Positioning System, is a global navigation system that uses satellites, a receiver and algorithms to synchronise location, velocity and time data for air, sea and land travel.

The GPS output is modelled as:

$$\begin{cases} y_{GPS,x} =^{I} p_x + \eta_{GPS,x}, & \eta_{GPS,x} \sim N(0, 3.24) \\ y_{GPS,y} =^{I} p_y + \eta_{GPS,y}, & \eta_{GPS,y} \sim N(0, 3.24) \\ y_{GPS,z} =^{I} p_z + \eta_{GPS,z}, & \eta_{GPS,z} \sim N(0, 0.6881) \end{cases} \quad (12)$$

The Gaussian was estimated based on data from Global Positioning System (GPS) Standard Positioning Service (SPS) Performance Analysis Report [6].

### 3.2 Distance Sensor

The distance sensor is an AI vision based sensor. As seen in the beginning of this section, the distance sensor is based on the relative positioning of the drones. In particular the module of the vectors. And its output is:

$$y_{distance} = ||\vec{d}_{i,j}|| + \eta_{distance}, [i \neq j] \in [1,2,3] \quad (13)$$

$$\eta_{distance} \sim N(0, 1 \times 10^{-6}) \quad (14)$$

The noise was arbitrated based on the article Error Evaluation in a Stereo-vision-Based 3D Reconstruction System [7].

### 3.3 Angle Sensor

The angle sensor is also an AI vision based sensor. The angle sensor is also based on the relative positioning of the drones. In particular the normalised vectors. And its output is:

$$y_{angle} = \frac{\vec{d}_{i,j}}{||\vec{d}_{i,j}||}, [i \neq j] \in [1,2,3] + \eta_{angle} \quad (15)$$

$$\eta_{angle} \sim N(0, 5 \times 10^{-7}) \quad (16)$$

The noise was arbitrated based on the article Error Evaluation in a Stereo-vision-Based 3D Reconstruction System [7].

## 4 Filtering

### 4.1 Kalman Filter

The kalman filter is an algorithm based on prediction, the first step, where it produces an estimate of the current state as well as its uncertainty and then a subsequent update, the second step.

The current state is calculated using the known system model, input vector as well as the previous state. This step does not include the system's process noise and nonlinearities. After observing the output measurements, which are corrupted with error, both the state and its uncertainty are updated, given more weight to the more certain estimates, this means that the algorithm is recursive.

It is assumed that the simulation errors and sensor's noise, $\eta_x$ and $\eta_y$, are Gaussian with zero mean and its variance can be obtained from experimental data. Also, it's assumed that the dynamic systems are linear and can be described in a matrix state space representation as follows:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + \eta_x(k), & \eta_x(k) \sim N(0, R_w) \\ y(k) = Cx(k) + Du(k) + \eta_y(k), & \eta_y(k) \sim N(0, R_v) \end{cases} \tag{17}$$

The variables above are, $k$, discrete time, $x$, state vector, $y$, output vector, $u$, input (or control) vector, $A$, state matrix, $B$, input matrix, $C$, output matrix, $D$, the feedforward matrix, $\eta_x$ and $\eta_y$, the process noise and sensor's error, are Gaussian with zero mean and $\sigma_x$ and $\sigma_y$ as variance.

The system's optimal state estimate is obtained using the system's predicted state estimate and the measurement with a weighted average. The kalman gain, evaluates which measures have smaller estimated uncertainty and therefore are trusted more.

The Measurement Update is defined as:

$$\begin{cases} \hat{x}(k+1|k+1) = \hat{x}(k+1|k) + \\ +K(k+1)[y(k+1) - C\hat{x}(k+1|k) - Du(k+1)] \\ \\ P(k+1|k+1) = P(k+1|k) - K(k+1)CP(k+1|k) \end{cases} \tag{18}$$

This process is repeated at every time step, with the new predicted state estimate and its covariance being dependent on the previous optimal state estimate. As seen in the following equation:

$$\begin{cases} \hat{x}(k+1|k) = A\hat{x}(k|k) + Bu(k) \\ P(k+1|k) = AP(k|k)A^T + R_w \end{cases} \tag{19}$$

The uncertainty of the measurements and of the current state estimate are fundamental to obtain the Kalman Gain, $K(k+1)$. The Kalman Gain is then used to calculate the optimal state estimate, equation (17).

With a high gain, the filter places more weight on the most recent measurements, and thus follows them more responsively. With a low gain, the filter follows the model predictions more closely.

$$K(k+1) = P(k+1|k)C^T[R_v + CP(k+1|k)C^T]^{-1} \tag{20}$$

### 4.2 Extended Kalman Filter

Previously, the Kalman Filter was described as a linear estimation algorithm. Once the first implementations were a non-linear one, an Extended Kalman Filter (EKF) must be implemented. The system is now represented as follows:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + \eta_x(k), & \eta_x(k) \sim N(0, R_w) \\ y(k) = h(x_k, u_k) + \eta_y(k), & \eta_y(k) \sim N(0, R_v) \end{cases} \tag{21}$$

The variables above are, $k$, discrete time, $x$, state vector, $y$, output vector, $u$, input (or control) vector, $A$, state matrix, $B$, input matrix, $h(x_k, u_k)$, measurement function, $\eta_x$ and $\eta_y$, the process noise and sensor's error, are Gaussian with zero mean and $\sigma_x$ and $\sigma_y$ as variance.

The predicted state estimate, $\hat{x}(k+1|k)$, is computed from the previous optimal state estimate, $\hat{x}(k|k)$, as in the Kalman Filter, and $h(x_k, u_k)$ is used to compute the predicted output, $\hat{y}(k+1|k)$, estimate using the predicted state estimate. Once $h(x_k, u_k)$ cannot be in a matrix form, the system must be linearized in each time step so that the Kalman Filter equations can be applied.

The linearization around the estimation is done computing, at each time step, the matrix of partial derivatives. The Jacobian is computed with the current predicted state estimation, $J(x(k+1|k), u(k+1)) \equiv J$.

$$J(x_k, u_k) = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_i} \\ \cdots & \cdots & \cdots \\ \frac{\partial h_j}{\partial x_1} & \cdots & \frac{\partial h_j}{\partial x_i} \end{bmatrix} \tag{22}$$

The Measurement Update is defined as:

$$\begin{cases} \hat{x}(k+1|k+1) = \hat{x}(k+1|k) + \\ +K(k+1)[y(k+1) - h(\hat{x}(k+1|k), u(k+1))] \\ \\ P(k+1|k+1) = P(k+1|k) - K(k+1)JP(k+1|k) \end{cases} \tag{23}$$

This process is repeated at every time step, with the new optimal estimate state and its covariance influencing the prediction of the next state estimation, as follows:

$$\begin{cases} \hat{x}(k+1|k) = A\hat{x}(k|k) + Bu(k) \\ P(k+1|k) = AP(k|k)A^T + R_w \end{cases} \tag{24}$$

Once again, the Kalman Gain affects the predicted state estimation. With a high gain, the filter places more weight on the most recent measurements, and thus follows them more responsively. With a low gain, the filter follows the model predictions more closely.

$$K(k+1) = P(k+1|k)J^T[R_v + JP(k+1|k)J^T]^{-1} \tag{25}$$

### 4.3 Kalman Filter as Complementary Filtering

Sensor fusion can be achieved by several algorithms, one of the less complex to implement is the complementary filter. The filter is defined by two gains that act as high and low pass filter to the sensors' output. This filter is specially important in the orientation estimation as the accelerometer is susceptible to vibrations that need to be filtered. In order to obtain a moving average, that is, a filtered acceleration, a low pass filter is the answer. On the opposite end of the spectrum, the rate gyro is accurate in the short term but due to the bias term the long term results lack accuracy. With this in mind a high pass filter is desired. That way the short-term gyroscope data is used while eliminating long term errors.

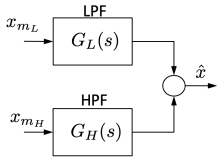


Fig. 4. Block Diagram.

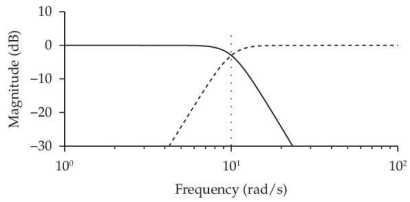


Fig. 5. Bode Diagram.

The filters are defined as:

$$G_L(s) = \frac{l}{s+l} \qquad G_H(s) = \frac{s}{s+l} \tag{26}$$

And the combination of both:

$$G_H(s) + G_L(s) = 1 \tag{27}$$

Figure 6 shows a scenario where the inputs are the rate gyro, $\dot{x}_{mH}$, and the magnetometer, $x_{mL}$. The filter is represented by $\hat{X}(s) = \frac{l}{s+l}X_{mL}(s) + \frac{s}{s+l}\frac{\dot{X}_{mH}(s)}{s}$.

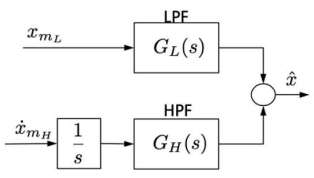


Fig. 6. Block Diagram of Complementary Filtering with integration.

The same filtering can be achieved using a Kalman Filter. The input is the rate gyro, $u = \dot{x}_{mH} = y_{gyro,z}$, and the output is the magnetometer, $y = x_{mL} = y_{mag}$. Recalling the Kalman equations:

$$\begin{cases} \dot{x} = Ex(t) + Fu(t) \\ y(t) = Gx(t) + Hu(t) \\ \dot{\hat{x}} = E\hat{x}(t) + Fu(t) + L(y(t) - G\hat{x}) \end{cases} \tag{28}$$

If equation (28) is rearranged, something similar to equation (27) appears:

$$\hat{Y}(s) = [G(sI - E + LG)^{-1}L]Y(s) + [G(sI - E + LG)^{-1}F]U(s) \tag{29}$$

$$\hat{X}(s) = \frac{GL}{sI - E + LG}X_{mL}(s) + \frac{GFs}{sI - E + LG}\frac{\dot{X}_{mH}(s)}{s} \tag{30}$$

### 4.4 Non-linear Filtering

The non-linear filter was designed to obtain the position estimation of drone 3. The state space model is defined as:

$$\begin{cases} \dot{x} = Ex(t) + Fu(t) \\ y(t) = h(x,t) \end{cases} \tag{31}$$

The variables above are, $t$, continuous time, $x$, state vector, $y$, output vector, $u$, input (or control) vector, $E$, state matrix, $F$ and $h(x,t)$ the output function.

For the stability proof of the non-linear filter the states are the position of drone 3, $\vec{p}_3$ and the control vector is the velocity of drone 3, $\dot{\vec{p}}_3$:

$$x = \vec{p}_3 \qquad u = \dot{\vec{p}}_3 \tag{32}$$

The inputs of the filter are the GPS positions of Drone 1 and Drone 2, as well as the angle sensor:

$$h^T = \begin{bmatrix} \vec{p}_1 & \vec{p}_2 & \frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||} & \frac{\vec{d_{2,3}}}{||\vec{d_{2,3}}||} \end{bmatrix} \tag{33}$$

Must be noted that:

$$\vec{d_{1,3}} = \vec{p}_3 - \vec{p}_1 \qquad \tilde{\vec{p}}_3 = \vec{p}_3 - \hat{\vec{p}}_3 \tag{34}$$

$$\frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||} \times (\vec{p}_3 - \vec{p}_1) = 0 \tag{35}$$

$$\hat{\vec{p}}_3 - \vec{p}_1 = -\tilde{\vec{p}}_3 + \vec{d_{1,3}} \qquad a \times b = S(a)b \tag{36}$$

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \times \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} a_2b_3 - b_2a_3 \\ b_1a_3 - a_1b_3 \\ a_1b_3 - b_1a_3 \end{bmatrix} = S(a) \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{37}$$

Using the previous equations:

$$S(\frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||})(\hat{\vec{p}}_3 - \vec{p}_1) = S(\frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||})(\hat{\vec{p}}_3 - \vec{p}_3 + \vec{p}_3 - \vec{p}_1) \tag{38}$$

$$S(\frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||})(-\tilde{\vec{p}_3}+\vec{d_{1,3}}) = -S(\frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||})\tilde{\vec{p}_3} \tag{39}$$

The non-linear filter takes the form:

$$\dot{\hat{\vec{p}}_3} = u + l\sum_{i=1}^{2} S(\frac{\vec{d_{i,3}}}{||\vec{d_{i,3}}||})^2(\hat{\vec{p}_3}-\vec{p}_i) \tag{40}$$

To check whether or not $\tilde{\vec{p}_3}$ converges asymptotically to zero, consider the error system:

$$\dot{\tilde{\vec{p}}_3} = \dot{\vec{p}_3} - \dot{\hat{\vec{p}}_3} \tag{41}$$

$$\dot{\tilde{\vec{p}}_3} = u - u + lS(\frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||})^2\tilde{\vec{p}_3} + lS(\frac{\vec{d_{2,3}}}{||\vec{d_{2,3}}||})^2\tilde{\vec{p}_3} \tag{42}$$

To prove that the error tends to zero Lyapunov stability arguments can be invoked, starting with a candidate Lyapunov function that is positive definite:

$$V(x) > 0 \tag{43}$$

$$V(x) = \begin{cases} 0, x = 0 \\ > 0, x \neq 0 \end{cases} \tag{44}$$

$$V(x) = \frac{1}{2}x^T x \tag{45}$$

The positive definiteness of $V(x)$ has been proven. If one can show that the time derivative of $V(x)$ is negative definite along the solutions of the system, then one guarantees that $x = 0$ is asymptotically stable.

$$\dot{V}(x) = \begin{cases} 0, x = 0 \\ < 0, x \neq 0 \end{cases} \tag{46}$$

$$\dot{V}(x) = x^T\dot{x} = \tilde{\vec{p}_3}^T l\sum_{i=1}^{2} S(\frac{\vec{d_{i,3}}}{||\vec{d_{i,3}}||})^2\tilde{\vec{p}_3} \tag{47}$$

Note that $S(a)T = -S(a)$. Therefore, $\dot{V} = -l\tilde{\vec{p}_3}^T\sum_{i=1}^{2} S^T(\frac{\vec{d_{i,3}}}{||\vec{d_{i,3}}||})S(\frac{\vec{d_{i,3}}}{||\vec{d_{i,3}}||})\tilde{\vec{p}_3}$ which is negative definite provided that $\vec{d_{1,3}}$ and $\vec{d_{2,3}}$ are not collinear.

When $\dot{V(x)} = 0$ when $x = 0$ and $x \to 0$ when $t \to \infty$ , therefore the filter guarantees convergence of $\hat{\vec{p}_3}$ to $\vec{p}_3$ and takes the form:

$$\dot{\hat{\vec{p}}_3} = u + l\sum_{i=1}^{2} S(\frac{\vec{d_{i,3}}}{||\vec{d_{i,3}}||})^2(\hat{\vec{p}_3}-\vec{p}_i) \tag{48}$$

## 5 Modeling

Each subsystem is defined by its equations, matrices and functions. The simulation subsystem can be described in state space representation in discrete time, as in equation (49) or equation (50).

A linear output is represented by:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + \eta_x(k), \eta_x(k) \sim N(0, R_w) \\ y(k) = Cx(k) + Du(k) + \eta_y(k), \eta_y(k) \sim N(0, R_v) \end{cases} \tag{49}$$

If the output is not linear it can be defined as:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + \eta_w(k), \eta_w(k) \sim N(0, R_w) \\ y(k) = h(x_k, u_k) + \eta_v(k), \eta_v(k) \sim N(0, R_v) \end{cases} \tag{50}$$

But, for a better intuitive understanding of the model, continuous time is used instead of discrete time.

$$\begin{cases} \dot{x} = Ex + Fu + \eta_w, & \eta_w \sim N(0, R_w) \\ y = Gx + Hu + \eta_v, & \eta_v \sim N(0, R_v) \end{cases} \tag{51}$$

As the block diagrams are expressed in discrete time, MATLAB functions can be used to transform the state space representation from continuous to discrete time.

### 5.1 Kalman Filter vs Complementary Filtering

In order to choose which implementation shall be used in the orientation (Kalman Filter or Kalman Filter as Complementary Filtering), a practical comparison between the two must be done.

The simulation is equal for both implementations and is defined by equation (51). Where the states are the heading and its rate of change:

$$x^T = \begin{bmatrix} \psi & r \end{bmatrix} \tag{52}$$

Once the orientation is a result of process error, $\eta_\psi$, the only input is the Bias, $u = \beta_{gyro,z}$.

$$E = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} F = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \eta_w = \begin{bmatrix} 0 \\ \eta_\psi \end{bmatrix} \tag{53}$$

The output is the magnetometer and the rate gyro:

$$G = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} H = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \eta_v = \begin{bmatrix} \eta_{mag} \\ \eta_{gyro} \end{bmatrix} \tag{54}$$

#### 5.1.1 Kalman Filter

Using the previous representation, the Kalman Filter is defined by heading, bias and heading rate of change as states.

$$x^T = \begin{bmatrix} \psi & \beta_{gyro,z} & r \end{bmatrix} \tag{55}$$

Once the orientation is a result of process error, $\eta_\psi$, there is no input.

$$E = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} F = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \eta_w = \begin{bmatrix} 0 \\ 0 \\ \eta_\psi \end{bmatrix} \tag{56}$$

The output is the magnetometer and the rate gyro:

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} H = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \eta_v = \begin{bmatrix} \eta_{mag} \\ \eta_{gyro} \end{bmatrix} \tag{57}$$

### 5.1.2 Kalman Filter as Complementary Filtering

Using the previous representation, the filter is defined by the heading and bias as the only states and the rate gyro as input of the filter, $u = y_{gyro,z}$.

$$x^T = \begin{bmatrix} \psi & \beta_{gyro,z} \end{bmatrix} \tag{58}$$

If the gyro equation is rearranged with $\dot{\beta}_{gyro,z} = 0$:

$$r = y_{gyro,z} - \beta_{gyro,z} - \eta_{gyro,z} \tag{59}$$

Using equation (59), the filter's matrices are the following, with the only output from the filter $\psi$.

$$E = \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix} F = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \eta_x = \begin{bmatrix} -\eta_{gyro} \\ 0 \end{bmatrix} \tag{60}$$

## 5.2 EKF - 3D Drone Positioning and Orientation

Any vector defined at any local referential can be transformed to the inertial referential and vice-versa:

$${}^{i}R_i^{V_i} p_j = {}^{I} p_j, \qquad {}^{I}R_i^{I} p_j = {}^{V_i} p_j \qquad i, j \in [1,2,3] \tag{61}$$

### 5.2.1 Simulation

The simulation takes the form of equation (51), but now, the measurements are set by $h(x,u) + \eta_v$, $\eta_v \sim N(0, R_v)$. The simulation states are the position of drone 1, the relative position of drone 1 and drone 3, as well as drone 3 and drone 2, and the respective velocities, plus the orientation.

$$x^T = \begin{bmatrix} \vec{p_1} & \vec{d_{1,3}} & \vec{d_{3,2}} & \phi_i & \theta_i & \psi_i & \dot{\vec{p}_1} & \dot{\vec{d}_{1,3}} & \dot{\vec{d}_{3,2}} & p_i & q_i & r_i \end{bmatrix} \tag{62}$$

The simulation inputs are, as before, the linear accelerations.

$$u^T = \begin{bmatrix} \vec{a_1} & \vec{a_2} & \vec{a_3} \end{bmatrix} \tag{63}$$

The simulation state space matrices are:

$$E = \begin{bmatrix} \mathbf{0}_{18} & I_{18} \\ \mathbf{0}_{18} & \mathbf{0}_{18} \end{bmatrix}_{36\times 36} \qquad F = \begin{bmatrix} & \mathbf{0}_9 & \\ & \mathbf{0}_9 & \\ I_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -I_3 & \mathbf{0}_3 & I_3 \\ \mathbf{0}_3 & I_3 & -I_3 \\ & \mathbf{0}_9 & \end{bmatrix}_{36\times 6} \tag{64}$$

The measurement function of the simulation is based on GPS, distance and angle sensor, rate gyro, accelerometer and magnetometer. It should be noted that the output of the angle sensor is expressed in the local referential. The output are , ${}^{O}h_{15}$, referent to the orientation and, ${}^{P}h_{15}$, referring to the positioning.

$${}^{P}h_{15}^T = \begin{bmatrix} \vec{p_1} & \vec{p_2} & ||\vec{d_{1,3}}|| & ||\vec{d_{3,2}}|| & ||\vec{d_{1,2}}|| & \frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||} & \frac{\vec{d_{3,2}}}{||\vec{d_{3,2}}||} & \frac{\vec{d_{1,2}}}{||\vec{d_{1,2}}||} \end{bmatrix} \tag{65}$$

$${}^{O}h_{15}^T = \begin{bmatrix} \psi_i & gyro_i & acc_i \end{bmatrix}, i \in [1,2,3] \tag{66}$$

### 5.2.2 Orientation

This filter has as states the orientation of the three drones, $[\phi_i, \theta_i, \psi_i], i \in [1,2,3]$, and the bias from the rate gyroscope, $\beta_i, i \in [1,2,3]$

$$x^T = \begin{bmatrix} \phi_i & \theta_i & \psi_i & \beta_{ix} & \beta_{iy} & \beta_{iz} \end{bmatrix} \tag{67}$$

Even though the orientation is a result of process noise, the system has an input, the rate gyro.

$$u^T = \begin{bmatrix} p_1 & q_1 & r_1 & p_2 & q_2 & r_2 & p_3 & q_3 & r_3 \end{bmatrix} \tag{68}$$

The state space matrices of the Extended Kalman Filter are:

$$E = \begin{bmatrix} \mathbf{0}_3 & -I_3 & \mathbf{0}_{3\times 12} \\ & \mathbf{0}_{3\times 18} & \\ \mathbf{0}_{3\times 9} & -I_3 & \mathbf{0}_{3\times 6} \\ & \mathbf{0}_{3\times 18} & \\ \mathbf{0}_{3\times 9} & \mathbf{0}_{3\times 6} & -I_3 \\ & \mathbf{0}_{3\times 18} & \end{bmatrix}_{18\times 18} \qquad F = \begin{bmatrix} I_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & I_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & I_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}_{18\times 9} \tag{69}$$

The output function is based on the magnetometer and the accelerometer. Recalling the respective equations, with $i \in [1,2,3]$:

$${}^{O}h^T = \begin{bmatrix} gsen(\theta_i), -gcos(\theta_i)sen(\phi_i), -gcos(\theta_i)cos(\phi_i), \psi_i \end{bmatrix} \tag{70}$$

And its Jacobian of the orientation measurement function, ${}^{O}h$, is:

$$J_O = \begin{bmatrix} \begin{bmatrix} 0 & a & 0 \\ b & c & 0 \\ d & e & 0 \\ 0 & 0 & 1 \end{bmatrix} & \mathbf{0}_{4\times15} & \\ \mathbf{0}_{4\times3} & \begin{bmatrix} 0 & f & 0 \\ h & k & 0 \\ l & m & 0 \\ 0 & 0 & 1 \end{bmatrix} & \mathbf{0}_{4\times12} \\ \mathbf{0}_{4\times6} & & \begin{bmatrix} 0 & n & 0 \\ p & q & 0 \\ r & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{0}_{4\times9} \end{bmatrix}_{12\times18} \tag{71}$$

$$a = gcos(\theta_1),\ b = -gcos(\theta_1)cos(\phi_1),\ c = gsin(\theta_1)sin(\phi_1) \tag{72}$$

$$d = gcos(\theta_1)sin(\phi_1),\ e = gsin(\theta_1)cos(\phi_1),\ f = gcos(\theta_2) \tag{73}$$

$$h = -gcos(\theta_2)cos(\phi_2),\ k = gsin(\theta_2)sin(\phi_2), \tag{74}$$

$$l = gcos(\theta_2)sin(\phi_2),\ m = gsin(\theta_2)cos(\phi_2),\ n = gcos(\theta_3) \tag{75}$$

$$p = -gcos(\theta_3)cos(\phi_3),\ q = gsin(\theta_3)sin(\phi_3), \tag{76}$$

$$r = gcos(\theta_3)sin(\phi_3),\ s = gsin(\theta_3)cos(\phi_3) \tag{77}$$

### 5.2.3 Position

In this iteration the states are the position of drone 1, $\vec{p}_1$, the distance between drone 3 and drone 1, $\vec{d_{1,3}}$, the distance between drone 3 and drone 2, $\vec{d_{2,3}}$ and the respective velocities, with coordinates $(x, y, z)$.

In continuous time the simulation is defined as follows:

$$x^T = \begin{bmatrix} \vec{p}_1 & \vec{d_{1,3}} & \vec{d_{3,2}} & \dot{\vec{p}}_1 & \dot{\vec{d}}_{1,3} & \dot{\vec{d}}_{3,2} \end{bmatrix} \tag{78}$$

The inputs are, as before, the linear accelerations.

$$u^T = \begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \vec{a}_3 \end{bmatrix} \tag{79}$$

The state space representation is:

$$\dot{x} = \begin{bmatrix} \mathbf{0}_3 & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}_{6\times6} x + \begin{bmatrix} \mathbf{0}_3 \\ \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix} \end{bmatrix}_{6\times3} u \tag{80}$$

In order to implement the Extended Kalman Filter the Jacobians must be defined. The three output functions are concatenations of three different sensors, GPS, distance and normalised distance vectors, therefore, the three Jacobians will also be a concatenation of three matrices.

The Jacobian of the GPS is:

$$J_{GPS} = \begin{bmatrix} I_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times9} \\ I_3 & I_3 & I_3 & \mathbf{0}_{3\times9} \end{bmatrix}_{6\times18} \tag{81}$$

The Jacobian of the distance sensor is:

$$J_{\vec{d_{i,j}}} = \begin{bmatrix} \mathbf{0}_3 & \begin{bmatrix} a_1 & b_1 & c_1 \\ 0 & 0 & 0 \\ g_1 & k_1 & l_1 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ d_1 & e_1 & f_1 \\ g_1 & k_1 & l_1 \end{bmatrix} & \mathbf{0}_{3\times9} \end{bmatrix}_{3\times18} \tag{82}$$

$$a_1 = \frac{d_{(1,3)x}}{||\vec{d_{1,3}}||}, \qquad b_1 = \frac{d_{(1,3)y}}{||\vec{d_{1,3}}||}, \qquad c_1 = \frac{d_{(1,3)z}}{||\vec{d_{1,3}}||} \tag{83}$$

$$d_1 = \frac{d_{(3,2)x}}{||\vec{d_{3,2}}||}, \qquad e_1 = \frac{d_{(3,2)y}}{||\vec{d_{3,2}}||}, \qquad f_1 = \frac{d_{(3,2)z}}{||\vec{d_{3,2}}||} \tag{84}$$

$$g_1 = \frac{d_{(1,2)x}}{||\vec{d_{1,2}}||}, \qquad k_1 = \frac{d_{(1,2)y}}{||\vec{d_{1,2}}||}, \qquad l_1 = \frac{d_{(1,2)z}}{||\vec{d_{1,2}}||} \tag{85}$$

The Jacobian of the normalised distance vectors is:

$$J_{\frac{\vec{d_{i,j}}}{||\vec{d_{i,j}}||}} = \begin{bmatrix} \mathbf{0}_3 & \begin{bmatrix} a_2 & b_2 & c_2 \\ b_2 & d_2 & e_2 \\ c_2 & e_2 & f_2 \end{bmatrix} & \mathbf{0}_{3\times12} \\ \mathbf{0}_{3\times6} & \begin{bmatrix} g_2 & k_2 & l_2 \\ k_2 & m_2 & n_2 \\ l_2 & n_2 & p_2 \end{bmatrix} & \mathbf{0}_{3\times9} \\ \mathbf{0}_3 & \begin{bmatrix} q_2 & r_2 & s_2 & q_2 & r_2 & s_2 \\ r_2 & t_2 & u_2 & r_2 & t_2 & u_2 \\ s_2 & u_2 & v_2 & s_2 & u_2 & v_2 \end{bmatrix} & \mathbf{0}_{3\times9} \end{bmatrix}_{9\times18} \tag{86}$$

$$a_2 = \frac{d^2_{(1,3)y} + d^2_{(1,3)z}}{||\vec{d_{1,3}}||^3},\ \ b_2 = -\frac{d_{(1,3)x} \times d_{(1,3)y}}{||\vec{d_{1,3}}||^3},\ \ c_2 = -\frac{d_{(1,3)x} \times d_{(1,3)z}}{||\vec{d_{1,3}}||^3} \tag{87}$$

$$d_2 = \frac{d^2_{(1,3)x} + d^2_{(1,3)z}}{||\vec{d_{1,3}}||^3},\ \ e_2 = -\frac{d_{(1,3)y} \times d_{(1,3)z}}{||\vec{d_{1,3}}||^3}, \tag{88}$$

$$f_2 = \frac{d^2_{(1,3)x} + d^2_{(1,3)y}}{||\vec{d_{1,3}}||^3},\ \ g_2 = \frac{d^2_{(3,2)y} + d^2_{(3,2)z}}{||\vec{d_{3,2}}||^3}, \tag{89}$$

$$k_2 = -\frac{d_{(3,2)x} \times d_{(3,2)y}}{||\vec{d_{3,2}}||^3}, \quad l_2 = -\frac{d_{(3,2)x} \times d_{(3,2)z}}{||\vec{d_{3,2}}||^3} \tag{90}$$

$$m_2 = \frac{d^2_{(3,2)x} + d^2_{(3,2)z}}{||\vec{d_{3,2}}||^3}, \quad n_2 = -\frac{d_{(3,2)y} \times d_{(3,2)z}}{||\vec{d_{3,2}}||^3} \tag{91}$$

$$p_2 = \frac{d^2_{(3,2)x} + d^2_{(3,2)y}}{||\vec{d_{3,2}}||^3}, \quad q_2 = \frac{d^2_{(1,2)y} + d^2_{(1,2)z}}{||\vec{d_{1,2}}||^3} \tag{92}$$

$$r_2 = -\frac{d_{(1,2)x} \times d_{(1,2)y}}{||\vec{d_{1,2}}||^3}, \quad s_2 = -\frac{d_{(1,2)x} \times d_{(1,2)z}}{||\vec{d_{1,2}}||^3} \tag{93}$$

$$t_2 = \frac{d^2_{(1,2)x} + d^2_{(1,2)z}}{||\vec{d_{1,2}}||^3}, \quad u_2 = -\frac{d_{(1,2)y} \times d_{(1,2)z}}{||\vec{d_{1,2}}||^3} \tag{94}$$

$$v_2 = \frac{d^2_{(1,2)x} + d^2_{(1,2)y}}{||\vec{d_{1,2}}||^3} \tag{95}$$

### 5.3 Non-linear Filter

#### 5.3.1 Simulation

In this iteration, the simulation states are the position of drone 1, $\vec{p}_1$, drone 2, $\vec{p}_2$, and drone 3, $\vec{p}_3$ plus the orientation $[\phi_i, \theta_i, \psi_i]$, with $i \in [1,2,3]$ and the respective velocities.

$$x^T = \begin{bmatrix} \vec{p}_1 & \vec{p}_2 & \vec{p}_3 & \phi_i & \theta_i & \psi_i & \dot{\vec{p}}_1 & \dot{\vec{p}}_2 & \dot{\vec{p}}_3 & p_i & q_i & r_i \end{bmatrix} \tag{96}$$

The simulation inputs are, as before, the linear accelerations.

$$u^T = \begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \vec{a}_3 \end{bmatrix} \tag{97}$$

The simulation state space matrices are:

$$E = \begin{bmatrix} \mathbf{0}_{18} & I_{18} \\ \mathbf{0}_{18} & \mathbf{0}_{18} \end{bmatrix}_{36\times 36} \qquad F = \begin{bmatrix} \mathbf{0}_9 \\ \mathbf{0}_9 \\ I_9 \\ \mathbf{0}_9 \end{bmatrix}_{36\times 6} \tag{98}$$

The measurement function of the simulation is based on GPS, angle sensor, rate gyro, accelerometer and magnetometer.

$${}^P h^T_{16} = \begin{bmatrix} \vec{p}_1 & \vec{p}_2 & \frac{\vec{d_{1,3}}}{||\vec{d_{1,3}}||} & \frac{\vec{d_{2,3}}}{||\vec{d_{2,3}}||} \end{bmatrix} \tag{99}$$

$${}^O h^T_{16} = \begin{bmatrix} \psi_i & gyro_i & acc_i \end{bmatrix}, i \in [1,2,3] \tag{100}$$

#### 5.3.2 Position

The goal is to obtain the estimation of Drone 3, using the non-linear filter. The position of drone 1 and 2, $\vec{p}_1$ and $\vec{p}_2$, is assumed to be the reading from the GPS sensors. The orientation filter is the same as in 5.2.2.

$$\dot{\hat{x}} = E\hat{x} + Fu + l\sum_{i=1}^{2} S(\frac{\vec{d_{i,3}}}{||\vec{d_{i,3}}||})^2(\hat{\vec{p}}_3 - \vec{p}_i) \tag{101}$$

The state is the position of drone 3 and its velocity. he control vector is the linear acceleration of drone 3.

$$x^T = \begin{bmatrix} \vec{p}_3 & \dot{\vec{p}}_3 \end{bmatrix} \qquad u = \vec{a}_3 \tag{102}$$

The matrices E, F and the gain, $l$, is the following matrix, with $a = 0.5$ and $b = 0.4$:

$$E = \begin{bmatrix} \mathbf{0}_3 & I_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}_{6\times 6} \qquad F = \begin{bmatrix} \mathbf{0}_3 \\ I_3 \end{bmatrix}_{6\times 3} \qquad l = \begin{bmatrix} aI_3 \\ bI_3 \end{bmatrix}_{6\times 3} \tag{103}$$

## 6 Results

The following table outlines the root mean square deviation in the two implementations tested to determine the heading, that is, Kalman Filter and Kalman Filter as Complementary Filtering.

| Configuration | RMS Deviation [º] |
|---|---|
| Kalman Filter | 0.611 |
| Complementary Filtering | 0.022 |

Table 1. Kalman Filter as Complementary Filtering - RMS Deviation Table.

Table 2 displays the root mean square deviation of the estimate and real position of Drone 3 in the EKF and Non-linear filters. Note that each local referential, $\{V_i\}$, $i \in [1,2,3]$, sympathetic with the drone's kinematic, is not only defined by its position, ${}^I p_i$, $i \in [1,2,3]$, but also, its orientation, $[\phi_i, \theta_i, \psi_i]$, $i \in [1,2,3]$, in an inertial referential, $\{I\}$.

| Config.: | $x-axis[m]$ | $y-axis[m]$ | $z-axis[m]$ |
|---|---|---|---|
| EKF | 0.8114 | 1.5697 | 1.0639 |
| Non-linear | 2.4023 | 2.3082 | 1.2903 |

Table 2. 15th Configuration - RMS Deviation Table of Drone 3.

## 7 Conclusions

The final configuration is a result of a series of implementations and simulations. During this process the deviation between the real position and the optimal estimate suffers alterations. Although in this paper only 3 configurations were tested, in the process of this conclusions fifteen configurations were tested.

The filter chosen to obtain the orientation was the Kalman Filter used as a Complementary Filter due to the better results shown in the tests. Recalling Table 1:

| Configuration | RMS Deviation [º] |
|---|---|
| Kalman Filter | 0.611 |
| Complementary Filtering | 0.022 |

Table 3. Orientation Filter - RMS Deviation.

As previously mentioned, sensor fusion can be achieved by several algorithms, one of the less complex to implement is the complementary filter. This filter is specially important in the orientation estimation as the accelerometer is susceptible to vibrations that need to be filtered. In order to obtain a moving average, that is, a filtered acceleration, a low pass filter is the answer. On the opposite end of the spectrum, the rate gyro is accurate in the sort term but due to the bias term the long term results lack accuracy. With this in mind a high pass filter is desired. That way the short-term gyroscope data is used while eliminating long term errors.

The first implementations were in two dimensions and with no orientation. It was observed that the same simulation, but instead of two dimensions tested in three dimensions, $(x,y,z)$, has substantially different results to the same sensors. In 3D, the best choice is the concatenation of both the distance and normalised vector between drones, therefore it was the configuration used in both the EKF and the Non-linear Filter. Recalling Table 2:

| Config.: | $x-axis[m]$ | $y-axis[m]$ | $z-axis[m]$ |
|---|---|---|---|
| EKF | 0.8114 | 1.5697 | 1.0639 |
| Non-linear | 2.4023 | 2.3082 | 1.2903 |

Table 4. 15th Configuration - RMS Deviation Table of Drone 3.

The introduction of the non-linear filter cannot be evaluated only based on the RMS deviation though. The need for less computing power as well as the need for less batteries due to the fewer sensors used and computing power can be a plus in some situations.

In order to evaluate the results some context on the environment in which the drones operate is fundamental. Nevertheless, both configurations tested are suited for implementation.

Future work should include the further development of the non-linear filter technique. For instance, the generalisation of this filter for all the estimates of all drones' positions and not only Drone 3. Also, for further validation, these configurations should be tested using an UAV on the context of real time navigation.

## 8 Bibliography


[1] Ide-Flore Kenmogne, Vincent Drevelle, Eric Marchand - Cooperative Localization of Drones by using Interval Methods - Acta Cybernetica, University of Szeged, Institute of Informatics, 2019, pp.1-16. hal-02339451. Available at `https://bit.ly/3bB79C1`.

[2] Kexin Guo, Zhirong Qiu, Wei Meng, Lihua Xie1 and Rodney Teo - Ultra-wideband based cooperative relative localization algorithm and experiments for multiple unmanned aerial vehicles in GPS denied environments. International Journal of Micro Air Vehicles [Em linha] vol. 9(3) 169–186, (2017), p.169-170. Available at `https://bit.ly/3bxey5p`.

[3] Matthew Leccadito - A Kalman Filter Based Attitude Heading Reference System Using a Low Cost Inertial Measurement Unit. Theses and Dissertations. Virginia Commonwealth University [Em linha] (2013) p.53 . Available at `https://bit.ly/3pVwZsZ`

[4] Matthew Leccadito - A Kalman Filter Based Attitude Heading Reference System Using a Low Cost Inertial Measurement Unit. Theses and Dissertations. Virginia Commonwealth University [Em linha] (2013) p.57 . Available at `https://bit.ly/3BHxZDc`

[5] Lasse Klingbeil, Christian Eling, Florian Zimmermann, and Heiner Kuhlmann - Magnetic Field Sensor Calibration for Attitude Determination. Journal of Applied Geodesy 8(2):97 -108 (2014). Available at `https://bit.ly/3vZSDNL`.

[6] Satellite Navigation Branch, ANG-E66 NSTB/WAAS T&E Team - GLOBAL POSITIONING SYSTEM STANDARD POSITIONING SERVICE PERFORMANCE ANALYSIS REPORT. October 2020. Available at `https://bit.ly/3pZZaH2`.

[7] Abdelkrim Belhaoua, Sophie Kohler and Ernest Hirsch - Error Evaluation in a Stereovision-Based 3D Reconstruction System.EURASIP Journal on Image and Video Processing (2010), p 8. Available at `https://bit.ly/3w7t2m1`.