A decentralized position observer for small-sat swarms exploiting GNSS and ranging measurements

Federico Oliva, Enrico Varriale, Domenico Cascone, Francesco Martinelli, Mario Sassano, Daniele Carnevale

***Abstract*—We propose a decentralized Kalman-based filter for absolute localization within a general swarm of small-sats, combining standard GNSS with inter-satellite ranging measurements. Differently from a centralized approach, each agent is responsible only for its own state estimation yet exploiting information from the rest of the fleet. The proposed approach reduces the uncertainty on the absolute position of more than one order of magnitude if compared to the case of GNSS-only measurement. The algorithm has been tested both on a complete and partial fleet topology. Results are provided by Monte Carlo simulations and supported by covariance and stability analysis.**

***Index Terms*—Swarm, Satellite, Ranging measurements, Localization, GNSS.**

In the last years, satellites have reached a considerably high technological level. Deployment and decommissioning are nowadays almost routine procedures. The launch of projects like Starlink and Galileo proved the control of satellite fleets to be a feasible and valuable path [1, 2]. These projects use very different satellites: *Starlink* works on LEO orbits and exploits small-sats, which are very different from the bigger and bulkier ones used by *Galileo* on MEO orbits. Indeed, small-sat swarms are a valid alternative to these last ones in many applications, like SAR missions [3]. Moreover, collaborative flight in swarms can be exploited to increase the spatial resolution of sensors. This kind of setup also brings advantages in terms of reliability and availability [4].

This work explicitly targets small-sat swarms to improve position estimation using only low-cost sensors, although the same methodology applies to bigger satellites and UAVs. Indeed, less precise sensors are usually installed on low-cost and low-space systems. Lastly, another reason for the small-sats choice is that they can be easily used to speed up the TRL development of a product in the so-called technological de-risking process. As reported in [5], nowadays, the leading standard in terms of small-sats is represented by CubeSats, which are usually deployed in *Low Earth Orbit* (LEO). CubeSats are highly used for research tasks, as described in [6] and [7]. For these reasons, the fleet considered in this work will consist of CubeSats.

As for satellite swarms, localization algorithms are of the utmost importance as they deliver position and attitude information, which is necessary for effective formation control. In the last decades, the most used algorithm for real-time position and attitude estimation of satellites has been the Kalman filter, along with all its improvements and modifications. Nonlinear systems are often addressed with versions like the extended Kalman filter (EKF) or the unscented Kalman filter (UKF) [8, 9]. Moreover, computational time performances and robustness can be significantly improved by exploiting solutions like the gain scheduling [10] and multiplicative Kalman filters [11]. Accuracy can also be increased through an online estimate of the process matrix noise, as described in [12].

Regarding position measurement, GNSS technology currently provides the most used absolute localization measure, with standard (non-differential) accuracy in the order of meters. Several technological solutions have been developed to increase the precision in the absolute localization, also on small-sat swarms. An excellent example is the DiGiTaL system (*Distributed Multi-GNSS Timing and Localization*), reaching a precision of centimeters [13]. As previously mentioned, multi-agent systems have been primarily investigated and used for UAV applications and small-sats. In this scenario, ranging sensors, e.g., Ultra Wide Band (UWB) transducers or Lidar-based sensors, have emerged as efficient ways of tackling distributed estimation and control tasks. Distributed control laws have been implemented to solve Leader-Following tasks, as described in [14], where the state estimation exploits consensus-based techniques. UWB sensors have been used in [15] for UAV 3D relative localization based on maximum likelihood estimation. In [16], sensor fusion of UWB and IMUs has been tested on UAV relative position estimate, exploiting a centralized Multiplicative Extended Kalman Filter (MEKF). The reader may refer to [17, 18, 19] for more information on Kalman-based algorithms and satellite state estimation. Lastly,

This work was partially funded by Thales Alenia Space and by the Italian Ministry for Research in the framework of the 2020 Program for Research Projects of National Interest (PRIN), under Grant 2020RTWES4. F. Oliva, F. Martinelli, M.Sassano and D. Carnevale are with the Dipartimento di Ing. Civile e Ing. Informatica, viale del Politecnico 00133, Rome, Italy, e-mail: federico.oliva@students.uniroma2.it, francesco.martinelli@uniroma2.it, daniele.carnevale@uniroma2.it, mario.sassano@uniroma2.it. E. Varriae and D. Cascone are with Thales Alenia Space, e-mail: enrico.varriale@thalesaleniaspace.com, domenico.cascone@thalesaleniaspace.com

range measurements have also been exploited on terrestrial rovers, as proposed in [20], while a more general sensor fusion approach is presented in [21]. In this work, relative distance measurements are provided to each agent through miniaturized antennas exploiting the UWB technology.

This work aims to propose a decentralized localization algorithm for absolute position estimation, with a low computational burden, exploiting ranging measures between agents in a fleet of small-sats. The agents are assumed to be 400m distant. We pre-process the GNSS data, exploiting relative distances and providing a new synthetic position measure to a Kalman-based observer. The algorithm has been implemented in a decentralized manner in order to speed up the computation and save transmission band. Note that we did not provide directly ranging measures to the Kalman filter since computational cost increases and the decentralization property is lost. As a matter of fact, the proposed approach does not add any considerable overhead to the standard Kalman filtering yet obtains a substantial improvement in the precision of the localization. Moreover, its decentralized structure opens the possibility of dealing with mesh-structured swarms, other than the classic star-structured ones considered in centralized algorithms. Lastly, the proposed approach has no strong technological assumptions, making it a suitable solution for several applications. Although the technologies used in this work are detailed as standard GNSS and UWB sensors, they could be easily replaced by other solutions, still yielding the validity of the proposed algorithm. These considerations describe the main contributions of this work, as the usual approach in localization algorithms is to make solid assumptions either on the sensors' technology or the communication model (e.g., [20, 16]). Instead, the proposed approach only details the noise characteristics of the measurements, which can be modified depending on the application.

The paper unfolds as follows; section I introduces both the CubeSat model used and the sensor setup for the fleet. Moreover, the standard Extended Kalman filter approach is revised. Section II presents the fleet communication model along with the decentralized position estimate algorithm. Furthermore, a covariance analysis for the algorithm is provided. In section III, the algorithm performances are compared through several metrics. The decentralized algorithm is tested against a Kalman filter with no relative distances. Section IV provides a stability analysis of the proposed algorithm to ensure the convergence of the estimation error. In section V conclusions and future work are presented.

I. Background

This section introduces the satellite model and the set of available sensors with their features. We also recall the Kalman filter theory at the end of the section.

A. System model

Consider a fleet of M CubeSats, which are miniaturized satellites made up by stacking multiple cubic modules of 10 cm side, weighting up to 40 kg [5]. However, the proposed methodologies can also be applied to larger/smaller satellites.

a) Orbital dynamics: consider the orbital dynamics of each satellite as in [18, 19], namely:

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{\|\boldsymbol{r}\|^3}\boldsymbol{r} + F_{drag}(\boldsymbol{r},\dot{\boldsymbol{r}}) + F_{J2}(\boldsymbol{r},\dot{\boldsymbol{r}}) + \frac{\mathbf{T}}{m} + \boldsymbol{w}(t), \tag{I.1}$$

where $\boldsymbol{r} \in \mathbb{R}^3$ is the spacecraft position with respect to the Earth-centered inertial system (ECI), μ is the gravitational constant, m is the mass of the satellite. The term F_{drag} models the drag effect caused by atmosphere friction, necessary for LEO satellites. The F_{J2} term models the second moment of the spherical harmonical representation (J2), affecting the orbit due to the Earth's oblation and gravity variability. $\mathbf{T}$ denotes the thrust provided by the actuation system, and it is assumed impulsive. In the context of this work, we let $\mathbf{T} \equiv 0$, namely, only the free evolution is considered. In fact, the input can influence the local observability property for nonlinear systems. Furthermore, both process and measurement noise have been taken into account as described in section I-B. As far as the process noise is concerned, $\boldsymbol{w}(t)$ has been modeled as Gaussian with zero mean and $\sigma_w = 1E-4$ m/s^2. Eq. (I.1) has been re-written in a standard state-space form [22]. Therefore, the state vector $\boldsymbol{x_o}$ and the input $\boldsymbol{u_o}$ are defined as

$$\boldsymbol{x_o} = \begin{bmatrix}\boldsymbol{r}^T & \dot{\boldsymbol{r}}^T\end{bmatrix}^T \in \mathbb{R}^6, \tag{I.2a}$$

$$\boldsymbol{u_o} = \mathbf{T} \in \mathbb{R}^3, \tag{I.2b}$$

where the o subscript stands for *orbit*, as this set of equations describes the orbital dynamics of the satellite.

B. Setup of available sensors

This paragraph describes the measurements available to each satellite. These measurements are expressed in the *Earth Centered Inertial* frame (ECI). Each satellite is provided with the following set of sensors:

- **GNSS:** GNSS technology is used to retrieve the absolute position of the satellite as well as a velocity estimation in the ECI reference frame. The measurement vector is modeled as

$$\boldsymbol{x}^{GNSS} = \begin{bmatrix}x, & y, & z, & \dot{x}, & \dot{y}, & \dot{z}\end{bmatrix}^T + \boldsymbol{d}^{GNSS} \in \mathbb{R}^6, \tag{I.3}$$

  where the vector $\boldsymbol{d}^{GNSS}$ is an additive uncertainty on the measurement process, and it is assumed as Gaussian distributed, namely

$$\boldsymbol{d}_v^{GNSS}: \quad \mu_v = 0 \text{ and } \sigma_v = 5 \text{ m } v \in \{1,\dots,3\}, \tag{I.4a}$$

$$\boldsymbol{d}_v^{GNSS}: \quad \mu_v = 0 \text{ and } \sigma_v = 2e^{-3}\ \frac{\text{m}}{\text{s}}\ v \in \{4,\dots,6\}. \tag{I.4b}$$

- **Ranging sensors:** Consider a fleet of M agents. It is assumed that each satellite measures the relative distances between agents, i.e., each satellite knows the adjacency matrix of the fleet (for more details, see section III-E). The fleet adjacency matrix is defined as

$$A = \begin{bmatrix} 0 & d_{12} & \dots & d_{1N} \\ d_{21} & 0 & \ddots & \vdots \\ \vdots & \ddots & 0 & d_{N-1N} \\ d_{N1} & \dots & d_{NN-1} & 0 \end{bmatrix}, \tag{I.5}$$

  where N is the number of agents communicating with the one performing the localization algorithm, with $N \leq M$. Therefore, generally speaking the communication is defined by a non-complete graph. The relative distance (RD) between agents i and j is defined as $d_{ji} = d_{ij} \triangleq ||\boldsymbol{r}_i - \boldsymbol{r}_j|| + d_r$, with d_r modeled as a Gaussian noise with the following characteristics:

$$d_r : \quad \mu_{RD,i} = 0 \text{ and } \sigma_{RD,i} = 0.2 \text{ m}. \tag{I.6}$$

  Those features have been obtained through a calibration process and measurement campaign on COTS UWB DECAWAVE technology ranging sensors.

Note that the satellite center of mass is slightly different from those of the GNSS and UWB sensors. This uncertainty has been neglected.

C. Packet loss model

The set algorithms described in this work have been tested, also considering sensor failures. More specifically, The temporary sensor failure has been modeled by a Gilbert-Elliot model [23] that captures a burst of failures. The main idea behind this model is to assign both a uniform *failure* and a *recovery* probability at each transmission. Every time a packet transmission is performed, it either succeeds or fails. If the failure happens, the next time the transmission is performed, there is no absolute certainty that it will succeed again. In fact, the previous failure could have been caused for instance by a sensor or network malfunctioning. Therefore, the result of the first transmission after a failure is sampled over the *recovery* distribution instead. As explained in [23], this process can be easily described by a two-state Markov chain. In this setup, the *recovery* and *failure* probabilities are defined for both GNSS and UWB sensors as follows:

- RD failure probability: 0.3
- RD recovery probability: 0.85
- GNSS failure probability: 0.3
- GNSS recovery probability: 0.95

D. Kalman filters for nonlinear systems

The standard for state and parameters estimation is the Kalman filter and all the algorithms based on it. Although it has been developed for a linear framework, several algorithms have been extended to work specifically with nonlinear systems. More particularly, the extended and unscented Kalman filters, which have been considered in this work, are often used for position, and attitude estimation [17]. Consider a generic nonlinear system in the state space form:

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{I.7a}$$
$$\boldsymbol{y}(t) = h(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{I.7b}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the system state, $\boldsymbol{u} \in \mathbb{R}^m$ is the input, and $\boldsymbol{y} \in \mathbb{R}^p$ is the system output. In the Kalman filtering framework, $f(\cdot)$ and $h(\cdot)$ are usually referred to as *transition* and *observation* models. In our case the transition model is represented by the pair of equations $\dot{x}_{o_1} = x_{o_2}$ and $\dot{x}_{o_2} =$ eq. (I.1). The observation model consists instead of the measurements defined in section I-B. If the system to be observed is described by (I.7) the linear Kalman filter can be extended as follows[17]:

1) **Extended Kalman filter:** the approach consists in linearizing the system along the trajectory in such a way that the well-known linear framework can be (locally) recovered, namely,

$$\dot{\boldsymbol{x}}(t) = \frac{\partial f(\boldsymbol{x}(t), \boldsymbol{u}(t))}{\partial \boldsymbol{x}(t)}\bigg|_{\bar{\boldsymbol{x}}} (\boldsymbol{x}(t) - \bar{\boldsymbol{x}}) + \frac{\partial f(\boldsymbol{x}(t), \boldsymbol{u}(t))}{\partial \boldsymbol{u}(t)}\bigg|_{\bar{\boldsymbol{u}}} (\boldsymbol{u}(t) - \bar{\boldsymbol{u}}), \tag{I.8a}$$
$$\boldsymbol{y}(t) = \frac{\partial h(\boldsymbol{x}(t), \boldsymbol{u}(t))}{\partial \boldsymbol{x}(t)}\bigg|_{\bar{\boldsymbol{x}}} (\boldsymbol{x}(t) - \bar{\boldsymbol{x}}) + \frac{\partial h(\boldsymbol{x}(t), \boldsymbol{u}(t))}{\partial \boldsymbol{u}(t)}\bigg|_{\bar{\boldsymbol{u}}} (\boldsymbol{u}(t) - \bar{\boldsymbol{u}}), \tag{I.8b}$$

   where the linearization point is defined as the pair $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})$. Then, the classic Kalman filter is applied to eq. (I.8). Clearly, this approach provides a local solution, and the linearization shall be computed any time the state moves significantly from its previous value [11]. If the EKF is applied on a discrete-time system, the linearization in eq. (I.8) shall be computed at every iteration on the current state estimate.
2) **Unscented Kalman filter:** this approach selects a set of points, referred to as *sigma points*, used to provide a statistical representation of the plant state. Then, the algorithm extracts the system covariance matrix from these points through a mathematical tool called unscented transform. From this information, the Kalman gain is computed (prediction step) and then used to estimate the state vector (correction step). Although it has good performance, this approach is more computationally demanding than the EKF.

E. Metrics of evaluation

In this section, we define the evaluation metrics used to assess the performance of the estimation algorithms. The position algorithm presented in section II returns a position value denoted as $\boldsymbol{p}_j$. In the forthcoming analysis, KF stands for a general Kalman Filter, specified in the next sections as EKF or UKF when necessary. The subscript defines the position measurement provided to the algorithm (either the GNSS or the $\boldsymbol{p}_j$), e.g., KF_{GNSS} stands for the KF provided with the GNSS position information, while KF_{RD} stands for the KF provided with $\boldsymbol{p}_j$. The performance of the algorithms has been interpreted through several indices based on the following position error definition:

$$\boldsymbol{e}_j(k) = (\boldsymbol{x}_j(k) - \hat{\boldsymbol{x}}_j(k)) \in \mathbb{R}^3, \tag{I.9}$$

for all $j \in \{1, \dots, M\}$ and $k \in \{1, \dots, N_{iter}\}$, where $\boldsymbol{x}_j(k) = \boldsymbol{r}_j(k)$ ($\boldsymbol{r}$ is defined in eq. (I.1)) and $\hat{\boldsymbol{x}}_j(k)$ is the estimated position. The following metrics are defined starting from the $j-th$ agent estimation error in eq. (I.9):

- Norm:

$$\delta_j(k) = \|\boldsymbol{e}_j(k)\| \in \mathbb{R}. \tag{I.10}$$

- Mean:

$$\delta_j^{\mu} = \frac{1}{N_{iter} - N_{ss} + 1} \sum_{k=N_{ss}}^{N_{iter}} \delta_j(k) \in \mathbb{R}, \tag{I.11}$$

- Standard deviation:

$$\delta_j^{\sigma} = \sqrt{\frac{1}{N_{iter} - N_{ss}} \sum_{k=N_{ss}}^{N_{iter}} (\delta_j(k) - \delta_j^{\mu})^2}. \tag{I.12}$$

  where $N_{ss} < N_{iter}$ is the number of samples after convergence.

II. Position Observer

In this section, we introduce a geometric approach merging GNSS data and relative distances among agents to evaluate the absolute position measurement of a single agent with improved precision compared to the GNSS-only case. The basic approach uses a Kalman-based filter to estimate the absolute position exploiting only GNSS sensor (position and velocities). On the contrary, we propose to combine GNSS position measurements with relative distances and a-priory estimates of agent positions, evaluated integrating eq. (I.1), in order to define a new synthetic measure that replaces the GNSS position used by the Kalman-based observer. Note that as we are exploiting relative measurements, no claim can be made on the accuracy improvement of the position estimate. It has to be remarked that each agent estimates only its own position. A comparative analysis is provided to show the effectiveness of the proposed approach with respect to the basic approach in the case of both EKF and UKF observers. As reported in section I-B, ranging sensors have higher precision than GNSS, although they give relative information rather than absolute. To wrap up, the idea behind the proposed algorithm is to combine a standard absolute measurement with relative ranging measurements characterized by lower noise. By doing so, the overall absolute position estimate improves in terms of precision.

A. Communication

The algorithm is decentralized, meaning that each satellite is responsible for its own position estimate. At each sampling time, the following information is assumed to be available for any j-th satellite of the fleet: GNSS, adjacency matrix A, current agent j a-priory position estimate $\boldsymbol{\chi}_{k|j}^{-}$, and $(N-1)$ a-priory position estimates $\boldsymbol{\chi}_{k|i}^{-}$ of the other agents. All the a-priory positions are wrapped in the $\boldsymbol{\chi}^{-}$ vector. Note that $j \neq i$, as j refers to the agent running the algorithm, while i to its neighbors. Therefore, the set of data available to each agent is the following:

$$\mathrm{I}_j = \left\{ \boldsymbol{x}_j^{GNSS},\, \boldsymbol{\chi}_{k|j}^{-},\, \boldsymbol{\chi}^{-},\, A \right\}. \tag{II.1}$$

The subscript k in $\boldsymbol{\chi}_{k|j}^{-}$ refers to the simulation sampling time index, while j to the agent running the algorithm, e.g. $\boldsymbol{\chi}_{3|2}^{-}$ describes the a-priory position of agent 2 at time 3.

B. Relative distance position algorithm

This paragraph presents the algorithm proposed to optimize and refine the agent's position estimate. The algorithm is based on a relatively intuitive geometric approach exploiting the ranging sensors' relative distance information. The procedure unfolds as follows:

1) **Interlink unity vector computation:** the first step consists in computing the unity vector between pairs of agents, namely

$$\hat{\boldsymbol{v}}_{ji} = \frac{\boldsymbol{\chi}_{k|i}^{-} - \boldsymbol{\chi}_{k|j}^{-}}{\left\| \boldsymbol{\chi}_{k|i}^{-} - \boldsymbol{\chi}_{k|j}^{-} \right\|}, \tag{II.2}$$

   where $\hat{\boldsymbol{v}}_{ji}$ is computed from the pair of a-priory position estimates $\boldsymbol{\chi}_{k|i}^{-}$ and $\boldsymbol{\chi}_{k|j}^{-}$.
2) **A-priory adjacency matrix:** consider all the a-priory absolute position estimates, namely the set $\boldsymbol{\chi}^{-}$ defined in eq. (II.1) and $\boldsymbol{\chi}_{k|j}^{-}$. From this set compute the a-priory adjacency matrix $A_{\boldsymbol{\chi}}$, filled with all the relative distances $d_{ji}^{\chi} \triangleq \left\| \boldsymbol{\chi}_{k|i}^{-} - \boldsymbol{\chi}_{k|j}^{-} \right\|$.
3) **Position estimate from relative distance:** the difference between measured and a-priory distance is projected along $\hat{\boldsymbol{v}}_{ji}$, namely,

$$\bar{\boldsymbol{\chi}}_{ji}^{-} \triangleq \boldsymbol{\chi}_{k|j}^{-} + \hat{\boldsymbol{v}}_{ji} \frac{d_{ji} - d_{ji}^{\chi}}{2}. \tag{II.3}$$

The $1/2$ factor is motivated by both agents correcting their initial position depending on the measured relative distance. Therefore the correction term is equally distributed between the two of them. By doing so, we compute the absolute position of agent j by exploiting the range measurement and a-priory position retrieved from agent i . Thus, each agent computes $(N-1)$ absolute position estimates from ranging measurements, namely the following set is introduced:

$$\bar{\boldsymbol{X}}_j^- = \bar{\boldsymbol{\chi}}_{ji}^- \ \cup_{i\in\{1,\dots,N\}\wedge i\neq j} . \tag{II.4}$$

4) **Centroid computation**: all the absolute positions estimated from the ranging measurements are summarized in a single value, namely the centroid of the set $\bar{\boldsymbol{X}}_j^-$:

$$\boldsymbol{\mathcal{C}}_j = \frac{\sum_i \bar{\boldsymbol{\chi}}_{ji}^-}{N-1}. \tag{II.5}$$

5) **Convex combination:** a convex combination of the triple $[\boldsymbol{\mathcal{C}}_j, \boldsymbol{x}_j^{GNSS}, \boldsymbol{\chi}_{k|j}^-]$ (see fig. 1) defines

$$\boldsymbol{p}_j = (1-\theta)\boldsymbol{\mathcal{C}}_j + \theta \boldsymbol{x}_j^{GNSS}, \ \theta \in [0,1]. \tag{II.6}$$

Note that in this computation $\boldsymbol{\chi}_{k|j}^-$ is contained in $\boldsymbol{\mathcal{C}}_j$.

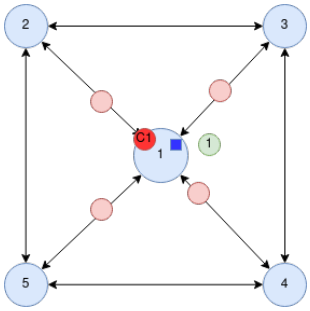


Figure 1: The a-priory estimates $\boldsymbol{\chi}^-$ of all the 5 agents are depicted in light blue. As for agent number 1, the absolute position estimates obtained from the ranging measurements $\bar{\boldsymbol{\chi}}_{ji}^-$ (highlighted in pink), the centroid $\boldsymbol{\mathcal{C}}_1$ in red, the $\boldsymbol{x}_1^{GNSS}$ measure in green. The blue square represents the new position measure $\boldsymbol{p}_1$.

Considering the new synthetic measure $\boldsymbol{p}$ in eq. (II.6), the output mapping changes from $h(\boldsymbol{x}, \dot{\boldsymbol{x}}) = [x \ y \ z \ \dot{x} \ \dot{y} \ \dot{z}]^T$ to $h(\boldsymbol{p}, \dot{\boldsymbol{x}}) = [p_x \ p_y \ p_z \ \dot{x} \ \dot{y} \ \dot{z}]^T$. It is important to stress such modification as it is involved in the Kalman gain evaluation. Note that $\boldsymbol{p}$ does embed an internal model, i.e. the copy of the small-sat plant in eq. (I.1), which implies that the estimation error will eventually converge to zero. However, we decided to combine $\boldsymbol{p}_j$ with a Kalman filter to exploit its disturbances filtering properties.

C. Position algorithm - Covariance analysis

This section analyses the precision of the newly defined $\boldsymbol{p}_j$ in eq. (II.6) according to the noise characteristics provided for GNSS and relative distance sensors. The goal is to show how $\boldsymbol{p}_j$ can increase the precision of the position estimation. Simulations will be compared with the theoretical expected results. This analysis is carried out in terms of measurement covariance matrices, following the well-known error propagation theory, which unfolds as follows:

1) Consider a generic set of variables $\{x_1, \dots, x_n\}$ with covariance matrix

$$\Sigma^x = \begin{bmatrix} \sigma_1^2 & \dots & \sigma_{1n} \\ \vdots & \ddots & \vdots \\ \sigma_{n1} & \dots & \sigma_n^2 \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{II.7}$$

2) Consider a generic map $\phi(\cdot) : \mathbb{R}^n \implies \mathbb{R}^m$
3) The covariance matrix of the function image is computed as follows:
   a) Consider the Jacobian of $\phi(\cdot)$

$$\mathbf{J} = \frac{\partial \phi}{\partial x} \in \mathbb{R}^{m\times n} \tag{II.8}$$

   b) The covariance matrix transformed through the map $\phi(\cdot)$ can be linearly approximated by

$$\Sigma^\phi = \mathbf{J}\Sigma^x\mathbf{J}^T \in \mathbf{R}^{m\times m} \tag{II.9}$$

The explicit computations of the covariance analysis are carried out on N agents. The agent performing its own position estimate is labelled as j, while all the others are indexed by i. In this analysis we consider $\boldsymbol{x}^{GNSS}$ as the position used to define $\hat{\boldsymbol{v}}_{ji}$ in eq. (II.2). This is done to compare $\boldsymbol{p}_j$ with GNSS regardless of any other filtering actions, such as Kalman filters. By doing so, we will evaluate the actual improvement of $\boldsymbol{p}_j$. Therefore, by following eq. (II.6), the map ϕ is

$$\phi = \frac{1-\theta}{N-1}\sum_{i\neq j}\bar{\boldsymbol{\chi}}_{ji}^- + \theta\cdot\boldsymbol{x}_j^{GNSS} \quad \phi(\cdot) : \mathbb{R}^{(3\cdot N+(N-1))} \implies \mathbb{R}^3. \tag{II.10}$$

The state covariance is

$$\Sigma^x = \begin{bmatrix} \sigma_{GNSS}^2\cdot\mathbf{I}_{3\cdot N} & 0 \\ 0 & \sigma_{RD}^2\cdot\mathbf{I}_{N-1} \end{bmatrix}, \tag{II.11}$$

with $\Sigma^x \in \mathbb{R}^{(3\cdot N+(N-1))\times(3\cdot N+(N-1))}$. The Jacobian $\mathbf{J}$ has the following structure:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial\phi_1}{\partial x_j} & \frac{\partial\phi_1}{\partial x_{i_1}} & \dots & \frac{\partial\phi_1}{\partial x_{i_{N-1}}} & \frac{\partial\phi_1}{\partial d_{ji_1}} & \frac{\partial\phi_1}{\partial d_{ji_{N-1}}} \\ \vdots & & \ddots & & & \vdots \\ \frac{\partial\phi_3}{\partial x_j} & \frac{\partial\phi_3}{\partial x_{i_1}} & \dots & \frac{\partial\phi_3}{\partial x_{i_{N-1}}} & \frac{\partial\phi_3}{\partial d_{ji_1}} & \frac{\partial\phi_3}{\partial d_{ji_{N-1}}} \end{bmatrix}, \tag{II.12}$$

with $\mathbf{J} \in \mathbb{R}^{3\times(3\cdot N+(N-1))}$.

D. Dispersion performance index

Now that both the KF-based and the geometric algorithms have been introduced, we define a different performance index that does not involve the estimation error norm but allows evaluating mean performance along the three axes. Since we desire to compare the EKF$_{GNSS}$ and EKF$_{RD}$ algorithms, we provide the following definitions where $\alpha \in \{"RD","GNSS"\}$:

- $j-th$ agent estimation error:

$$\boldsymbol{e}_j^{\alpha}(k) = (\boldsymbol{x}_j^{\alpha}(k) - \hat{\boldsymbol{x}}_j^{\alpha})(k) \in \mathbb{R}^3, \tag{II.13a}$$

  where both (j,k) follow the same definition of eq. (I.9).
- $j-th$ agent mean estimation error over the sample window $[N_{ss}, N_{iter}]$ along each dimension $i \in \{x,y,z\}$:

$$\mu_{ji}^{\alpha} = \frac{1}{N_{iter} - N_{ss} + 1} \sum_{k=N_{ss}}^{N_{iter}} \boldsymbol{e}_j^{\alpha}(k,i) \in \mathbb{R}. \tag{II.14a}$$

- $j-th$ agent estimation error standard deviation over the sample window $[N_{ss}, N_{iter}]$ along each dimension $i \in \{x,y,z\}$:

$$\sigma_{ji}^{\alpha} = \sqrt{\frac{1}{N_{iter} - N_{ss}} \sum_{k=N_{ss}}^{N_{iter}} (\boldsymbol{e}_j^{\alpha}(k,i) - \mu_{ji}^{\alpha})^2} \in \mathbb{R}. \tag{II.15a}$$

- $j-th$ agent estimation error standard deviation over the three dimensions:

$$\sigma_j^{\alpha} = \frac{1}{3} \sum_{i=1}^{3} \sigma_{ji}^{\alpha}. \tag{II.16}$$

- Ratio of standard deviations to compare precision of EKF$_{GNSS}$ and EKF$_{RD}$ for the $j-th$ agent:

$$\eta_j^{\sigma} = \frac{\sigma_j^{\text{RD}}}{\sigma_j^{\text{GNSS}}}, \tag{II.17}$$

- Fleet algorithm precision comparison:

$$\boldsymbol{\Gamma} = [\eta_1^{\sigma}, \ldots, \eta_N^{\sigma}]^T. \tag{II.18}$$

  The lower the ratio, the higher the precision of EKF$_{RD}$ compared to EKF$_{GNSS}$ and vice versa.

This index aims to highlight the mean standard deviation improvement using $\boldsymbol{p}_j$ with respect to GNSS.

III. Simulation results

This section presents the simulation results of the Kalman filters and the position observer described in section II. The goal of the section is to show the effectiveness of the proposed approach via extensive numerical simulations of eq. (I.1), where the estimated position is evaluated by a Kalman Filter (KF) fed with $\boldsymbol{p}_j$ in place of the absolute positions from the GNSS. Recall that the GNSS also provides velocity measurements in input to the KF. The model on which the Kalman filter has been designed is the one described in eq. (I.1) (with no process noise), namely the same used for the system evolution within the simulations. The section firstly shows results related to the covariance analysis presented in section II-C. Then, it proceeds comparing the performances of the EKF and UKF, in order to select the best KF-based algorithm on top of which we apply the geometric algorithm proposed in section II. This last analysis follows the approach described for attitude estimation in [9]. Therefore, no ranging measurements are used in section III-B. Then, in section III-C, the position estimate $\boldsymbol{p}_j$ in eq. (II.6) is added to the Kalman filtering, highlighting its precision improvements. The simulations have been run considering $N = 4$ agents, a sampling time of $T_s = 1\ s$, and $N_{iter} = 500$ iterations. The initial position error of the fleet is nearly 500 m for each agent. This error is substantial, but it has been used to highlight the convergence of the algorithms. The metrics used to assess the performance of the algorithms are those introduced in section I-E. All the results presented in the next sections come from a batch of 200 Monte Carlo runs, where the agents' initial position and the *classical orbit elements* were randomly initialized. The measurement noise was generated by changing the random seed at each simulation. The performance indices are then averaged over the whole set of Monte Carlo simulations (after convergence).

A. Covariance analysis

This section presents the results in terms of covariance, following the study reported in section II-C. We show that the j-th agent filtered position $\boldsymbol{p}_j$ defined in eq. (II.6) has a better precision with respect to the GNSS even if in place of the estimated position $\boldsymbol{\chi}$ we consider $\boldsymbol{x}^{GNSS}$ to define $\hat{\boldsymbol{v}}_{ji}$. We consider the GNSS measurements in eq. (II.6) to have an analytical solution of the Jacobian considered in section II-C. Simulations are performed integrating eq. (I.1) to compute the orbit. Measurements are computed from the integration result and are subject to added noise coherently to the specifications defined in section I-B. The theoretical covariance values obtained with the result of section II-C are compared to the average covariance over the whole simulation time (after convergence).

This comparison has been done for several θ values and $N = 2, N = 4, N = 6$ agents. This was done to show that an improvement in localization precision occurs as θ vanishes and the number of agents increases, as suggested by eq. (II.10). These results are reported in fig. 2. Note that the precision improvement differs between the three axes. We chalk this up to the swarm formation. Indeed it is reasonable to think that same number of agents in different formations results in different precision improvements. This is caused by the sensitivity of the trajectory to a specific

axis with respect to the others. We give an example of this phenomenon in fig. 3, where we consider $N = 4$ agents in two different formations. As you can see in fig. 3, the precision improvements are different. Therefore, these considerations open up a very interesting problem, namely the choice of the formation that maximizes the estimation precision. We are willing to investigate this issue in future works.

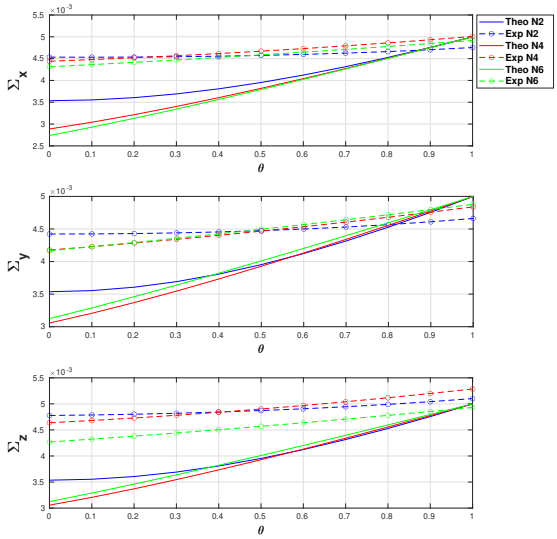


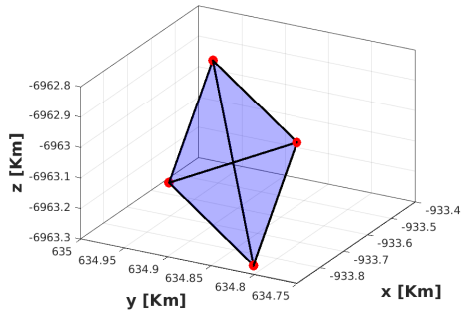
Figure 2: The figure shows the covariance analysis depending on the number of agents M and parameter θ. Solid lines are the theoretical covariance values expected on the considered trajectory (eq. (II.10)), while the dashed show σ averaged over the actual simulation data after convergence (eq. (II.9)).

Regarding the setup, all the simulations last 200 s each. The analytical expression of the covariance matrix Σ^{ϕ} has been evaluated on the true trajectory, namely $\forall \boldsymbol{x}_j,\ j \in \{1, \dots, M\}$. Note that the covariance matrix Σ^{ϕ} has off-diagonal entries that are smaller than the diagonal ones by one order of magnitude. For this reason, we have approximated the (total) theoretical covariance along the three axes with its diagonal terms.

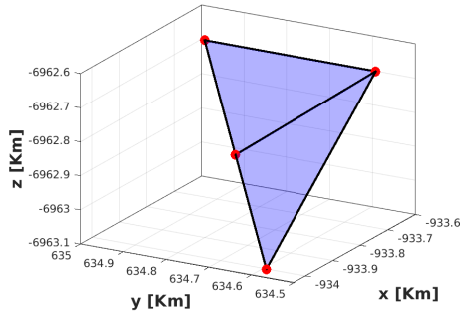
B. EKF$_{GNSS}$ VS UKF$_{GNSS}$

This section compares the simulation results of both EKF$_{GNSS}$ and UKF$_{GNSS}$, namely the Kalman filters provided with the GNSS information. Note that these filters are applied on the single agent without exploiting the ranging measurements. In the standalone simulation, EKF tends to perform slightly better, as shown by the boxplots in fig. 4 and in table I, where the performance of the algorithms is compared in terms of δ_j^{μ} and δ_j^{σ}.

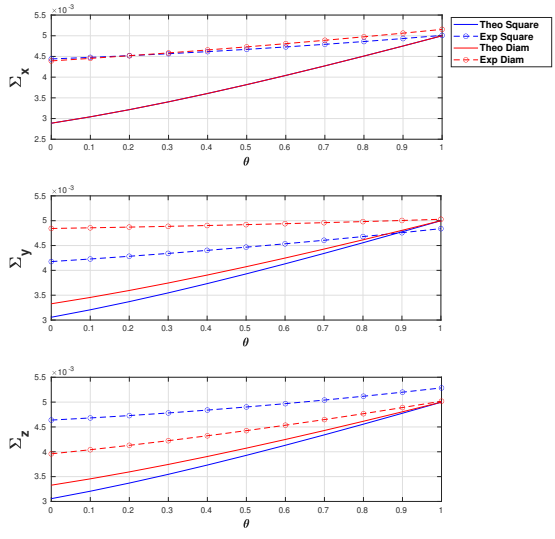
The EKF$_{GNSS}$ estimates the satellite position with a mean error norm of nearly 6.5m, while the UKF$_{GNSS}$ almost 8m,



(a) Square formation



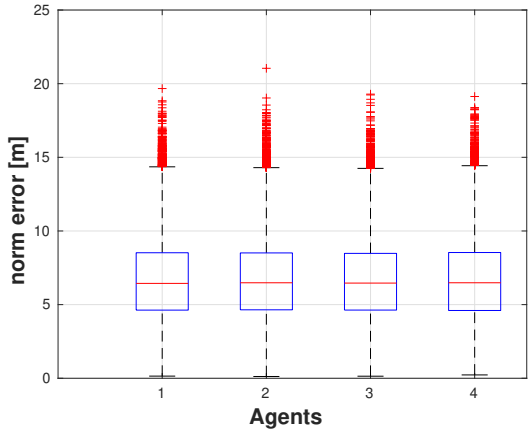
(b) Diamond formation



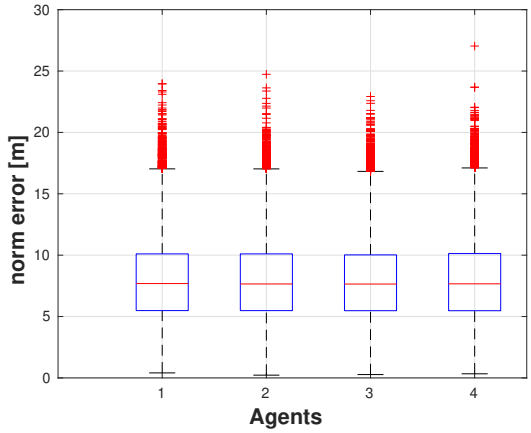
(c) Covariance analysis

Figure 3: The first two figures show the swarm structure (ECI coordinates). The third one shows the covariance analysis depending parameter θ. Solid lines are the theoretical covariance values, while the dashed show σ averaged over the actual simulation data after convergence (eq. (II.9)).

which is approximately $\sqrt{3}\sigma_i^{GNSS}$ as defined in section I-B.



(a) EKF$_{GNSS}$ boxplot



(b) UKF$_{GNSS}$ boxplot

Figure 4: Boxplot for both EKF$_{GNSS}$ and UKF$_{GNSS}$. The red horizontal lines show the **median** of the norm estimation errors for each agent. The box shows the 25 and 75 quartiles, and the red crosses are the outliers. The two boxplots are presented in different subplots to highlight the fact that the position of each agent in the formation does not affect, in general, the precision of the estimation.

The poor performances of the UKF compared to the EKF can be explained following the results presented in [9], even though this work addresses position rather than attitude estimation. More specifically, the EKF is more robust than the UKF to the initial condition provided to the filter. This results in a poorer convergence of the latter over a batch of simulations rather than on a single one, which is the case in this work[1]. The UKF$_{GNSS}$ has also a higher mean computational time[2] compared to EKF, namely 0.44 s vs 0.063 s . The sigma points propagation causes this slow-down effect due to the system dynamics integration, which requires more time. Based on these results, for the forthcoming analysis, we select the EKF$_{GNSS}$ as our standard Kalman filter, either provided with GNSS or $\boldsymbol{p}_j$.

C. EKF$_{RD}$ - tuning of the parameter θ and improvements using p_j

This section focuses on the effect of the parameter θ in eq. (II.6), on the estimation performance of EKF_{RD}. In the first iterations of EKF$_{RD}$ we consider the GNSS measurement instead of $\boldsymbol{p}_j$ since the unity vectors $\hat{\boldsymbol{v}}_{ji}$ depend on $\boldsymbol{\chi}_j$, which is initially affected by a large position error. Therefore, the GNSS measure is used until the a-priory position estimate error reaches an acceptable precision $(\pm 2\sigma_{GNSS})$, i.e., about five iterations[3]. After this transient $\boldsymbol{p}_j$ is considered. The overall performance comparison between EKF$_{RD}$ and EKF$_{GNSS}$ is shown in the boxplot fig. 5 and in table II. Even if the communication doesn't need to be defined by a complete graph, as reported in section I-B, we assumed it in the following simulations. This was done because the sensors are assumed to work in broadcast and on a multi-hop mesh network structure. For a detailed analysis of the results in terms of standard deviation, see section III-D.

A key point in the EKF is the computation of the (linearized) output mapping matrix, namely the Jacobian of the observation model:

$$H = \left.\frac{\partial h(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\bar{\boldsymbol{x}}}. \tag{III.1}$$

As reported in section III-A, if $\boldsymbol{p}_j$ is used, the output mapping is

$$h(\boldsymbol{p}_j, \boldsymbol{x}) = [p_{j,x}\ p_{j,y}\ p_{j,z}\ x^{GNSS}_{\dot{x}}\ x^{GNSS}_{\dot{y}}\ x^{GNSS}_{\dot{z}}] \tag{III.2}$$

[1]We consider convergence reached when the standard deviation of the estimation error over a buffer of samples remains within $\pm\sigma$ of the measurement noise assumptions.

[2]Tested on a *11th Gen Intel(R) Core(TM) i7-11700 @ 2.50GHz*

[3]Clearly this transient was identified offline as the estimation error is not available during the simulation

N. Agent	EKF$_{GNSS}$ δ_j^{μ} [m]	UKF$_{GNSS}$ δ_j^{μ} [m]	EKF$_{GNSS}$ δ_j^{σ} [m]	UKF$_{GNSS}$ δ_j^{σ} [m]
Agent 1	6.70	7.96	2.83	3.36
Agent 2	6.71	7.95	2.81	3.36
Agent 3	6.67	7.92	2.78	3.34
Agent 4	6.71	7.96	2.81	3.38

Table I: Algorithms comparison in terms of δ_j^{μ} and δ_j^{μ} (m). The EKF performs better slightly better than the UKF. Both algorithms are provided with GNSS measurements.

N. Agent	EKF$_{GNSS}$ δ_j^{μ} [m]	EKF$_{\boldsymbol{RD}}$ $\boldsymbol{\delta_j^{\mu}}$ [m]	EKF$_{GNSS}$ δ_j^{σ} [m]	EKF$_{\boldsymbol{RD}}$ $\boldsymbol{\delta_j^{\sigma}}$ [m]
Agent 1	6.70	**0.75**	2.83	**0.33**
Agent 2	6.71	**0.74**	2.81	**0.33**
Agent 3	6.67	**0.75**	2.78	**0.33**
Agent 4	6.71	**0.72**	2.81	**0.31**

Table II: Estimation performance in terms of δ_j^{μ} and δ_j^{μ} (m). EKF_{RD} outperforms EKF_{GNSS} of about one order of magnitude.

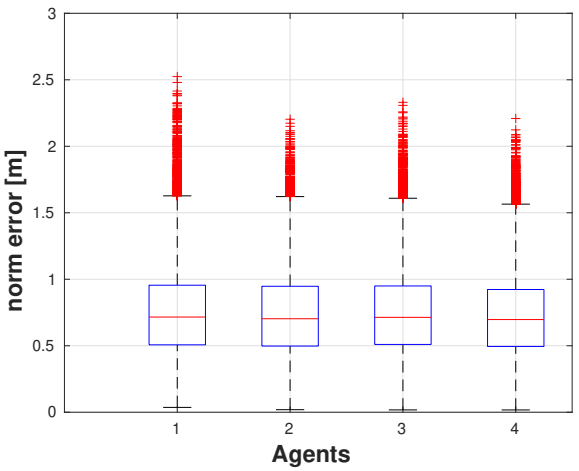


Figure 5: Boxplot for EKF$_{RD}$: red horizontal lines are the **median** of the norm estimation errors for each agent. The box shows the 25 and 75 quartiles and the red crosses are the outliers.

where $p_{(j,\cdot)}$ describes the projection of the new position estimate along each axis, while $x^{GNSS}_{\dot{x},\dot{y},\dot{z}}$ are the orbital velocity components given by GNSS as described in section I-B. In order to correctly use the EKF, the Jacobian of the observation model assumes the following form:

$$H_{\boldsymbol{p}} = \begin{bmatrix} \dfrac{\partial \boldsymbol{p}}{\partial \boldsymbol{x}} & 0 \\ 0 & I_{3\times 3} \end{bmatrix}, \tag{III.3}$$

which is the form we considered in the simulations. As for parameter θ, the higher its value, the more the GNSS measures are weighted to define $\boldsymbol{p}_j$, and the lower the RD are used. Coherently with the results presented in section III-A, we have seen that simulations with values of θ close to 0 provided the best estimation performance. However, small values of θ yield a slower convergence, as the a-priory position is weighted more. Furthermore, a dynamic selection of θ would improve convergence properties as shown in fig. 6: a higher value of θ is profitable initially to fasten the estimation error convergence. In contrast, lower values are desirable for lower steady-state estimation error. The former consideration led us to propose a further refinement: a dynamic selection of θ by looking at the condition number of the state covariance matrix computed in the EKF, namely

$$k(P_x) = \frac{\min \sigma(P_x)}{\max \sigma(P_x)}, \tag{III.4}$$

$P(x)$ being the estimated covariance matrix. Since $k(P_x)$ decreases as the estimation improves, we select $\theta = 0.03$ if $|\mathrm{d}k(P_x)/\mathrm{dt}| \geq 1E^{-8}$, and $\theta = 0.005$ otherwise (These values were selected through a grid search in the interval $\theta \in [0, 1]$). The effect of this dynamic selection of θ is shown as the yellow dotted line in fig. 6. In terms of convergence time, the dynamic θ configuration requires nearly 300s to reach the steady state estimation error.

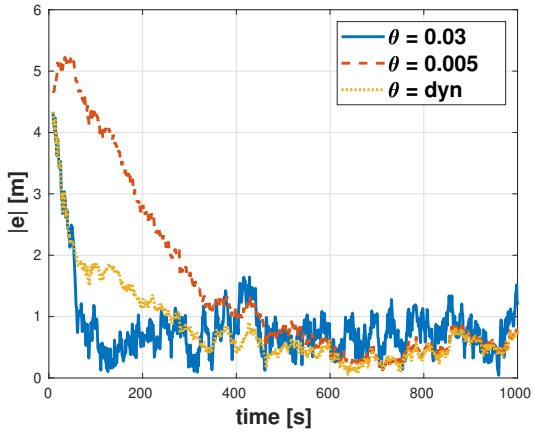


Figure 6: Effect of θ on the norm of the estimation error δ_j of a single agent: in blue solid line $\theta = 0.03$ (faster convergence lower precision) and in red dashed line $\theta = 0.005$ (slower convergence higher precision). In yellow dotted line a dynamic selection of θ.

Lastly, it shall be noted that the proposed geometric algorithm requires on average 1.3ms, adding almost no overhead to the standard Kalman filtering. This low computational burden makes the proposed solution appealing compared to more complex sensor fusion solutions (e.g., SLAM or centralized Kalman filtering).

D. Dispersion analysis

This short paragraph aims to compare the results of EKF$_{GNSS}$ and EKF$_{RD}$ in terms of their dispersion, as described in section II-D. Indeed, GNSS and ranging measurement noise are modelled with zero mean errors and then it is essential to provide such an index. Certainly, EKF$_{GNSS}$ and

EKF_{RD} have comparable mean values close to zero. Still, the filtering action on the EKF_{RD} reduces the standard deviation by more than an order of magnitude, given that the Monte Carlo simulations yield:

$$\boldsymbol{\Gamma} = \mathbf{10^{-2}} \begin{bmatrix} 9.62 & 9.63 & 9.53 & 9.44 \end{bmatrix}^T. \tag{III.5}$$

E. Performance analysis in case of missing measurements

The EKF_{RD} algorithm has also been tested in the eventuality that a GNSS or ranging measurement is missing. This analysis mimics a communication described by a non-complete graph, where the flawed communication follows the model presented in section I-C. Note that in case of broken communication between two agents (i, j), agent j loses both the relative distance measurement d_{ji} and the a-priory position estimate $\boldsymbol{\chi}^-_{k|i}$. As for the simulation setup, we considered a batch of 50 Monte Carlo simulations, each of them 500s long. The failures could occur as soon as the simulation started, accordingly to the probabilities detailed in section I-C. As for the θ values, we considered the same configuration proposed in section III-C. The simulation results are reported in table III. Here, the performance metrics δ^{RD} and σ^{RD} are averaged over the number of agents, namely

$$\bar{\delta} = \frac{1}{N}\sum_j \delta^{RD}_j, \qquad \bar{\sigma} = \frac{1}{N}\sum_j \sigma^{RD}_j \tag{III.6}$$

Clearly in case of temporary sensor failures the estimation performance worsen, but are still such that $\bar{\delta}_{fail} + 2\bar{\sigma}_{fail} < \sigma_{GNSS} = 5\text{m}$ (see section I-B).

Transmission	$\bar{\delta}$ [m]	$\bar{\sigma}$ [m]
No failures	0.74	0.30
With failures	0.78	0.34

Table III: Performance comparison (m) in case of missing measurements. This analysis considers again four agents in the fleet. Results are still on a set of Monte Carlo simulations.

IV. Stability analysis of the proposed algorithm

In section III-A we provided a quantitative analysis of the estimation improvement yield by using $\boldsymbol{p}_j$. In such analysis, $\boldsymbol{p}_j$ has been evaluated with the GNSS measures in place of the estimates $\boldsymbol{\chi}_j$. Therefore, no Kalman filter has been considered. Later, in section III-B the combined usage of $\boldsymbol{p}_j$ and a Kalman filter has been shown to improve the position estimation in terms of precision compared to single GNSS measurements. However, when the estimates $\boldsymbol{\chi}$ of all agents provided by EKF are used to define $\boldsymbol{p}_j$, together with the ranging measurements, there is a closed-loop in the estimation error dynamics, and therefore a stability analysis is necessary. More specifically, the closed-loop can be summarized as follows:

1) the a-priory position value is obtained through the propagation of the transition model in eq. (I.1).
2) sensor data (GNSS and UWB) are obtained and provided to the position algorithm.
3) the position filter in eq. (II.6) uses data from the current agent and from all its neighbors to compute a new set of measured data ($\boldsymbol{p}_j$) through the observation model (eq. (III.3)).
4) the new a-priory position value is corrected through the Kalman filter considering both the previous a-priory knowledge and $\boldsymbol{p}_j$.

Clearly, step 3 links the proposed algorithm to the Kalman filter through $\boldsymbol{p}_j$. If step 3 were removed, the standard Kalman-based approach would be implemented. Indeed, such approach has been proven to guarantee the convergence of the estimation error dynamics. In general, we cannot state the same if an additional filter is used, like the one proposed in this paper. Hence, the necessity to develop a stability analysis of the very same.

Proposition 1: Consider an EKF observer to estimate the state $\boldsymbol{x_{o,j}}$, within a subset $\Omega_\rho \triangleq \{\boldsymbol{x_{o,j}} \in \mathbb{R}^6 : ||\boldsymbol{x_{o,j}} - \bar{\boldsymbol{x}}_{\boldsymbol{o},j}|| \leq \rho\}$, whose evolution is given by eq. (I.1) that describes the orbital dynamics of the $j-th$ satellite around $\bar{\boldsymbol{x}}_{\boldsymbol{o},j}$. Assume that the position measure in the EKF is $\boldsymbol{p}_j$ as defined in eq. (II.6), whereas velocity measure is provided by GNSS, i.e. the input injection matrix of the EKF is given by (III.3). Furthermore assume that fleet configuration is such that $d_{ji} > 0$. Then there exist sufficiently small $\rho > 0$, sensor measurement disturbance amplitude $\bar{d}$ and $\gamma > 0$, where $\gamma = 1 - \theta$ with $\theta \in [0,\ 1]$ as in eq. (II.6), such that the origins of the position estimation error dynamics for all agents are exponentially asymptotically stable.

Proof 1: Let $\boldsymbol{e}_j = \boldsymbol{x_{o,j}} - \hat{\boldsymbol{x}}_{\boldsymbol{o},j}$ be the position estimation error of the $j-th$ satellite and let $W_0(\boldsymbol{e}_j, x_{\boldsymbol{o},j})$ be the estimation error dynamics if the measurement vector is given by GNSS coordinates, i.e. $\gamma = 0$ ($\theta = 1$ in eq. (II.6)). Therefore,

$$\dot{\boldsymbol{e}}_j = W_0(\boldsymbol{e}_j, \boldsymbol{x_{o,j}}, d), \tag{IV.1}$$

where d represents the measurement noise vector. Note that $W_0(\cdot,\cdot,\cdot)$ is smooth given LEO satellite's orbits, i.e. $r >> 0$ in eq. (I.1), and its linearization in $(\boldsymbol{e}_j, \boldsymbol{x_{o,j}}) = (0, \bar{\boldsymbol{x}}_{\boldsymbol{o},j})$ is asymptotically stable with respect to d which, by linearity, implies that $(0, \bar{\boldsymbol{x}}_{\boldsymbol{o},j})$ is locally exponentially stable with $d \equiv 0$. Consider now the case in which the GNSS position is replaced by $\boldsymbol{p}_j$ that can be equivalently rewritten as

$$\boldsymbol{p}_j = \gamma \boldsymbol{\mathcal{C}}_j + (1-\gamma)\boldsymbol{x}^{GNSS}_{\boldsymbol{o},j}, \tag{IV.2}$$

that is, $\boldsymbol{p}_j = \boldsymbol{x}^{GNSS}_j$ if $\gamma = 0$. Then, by smoothness of terms involved in eq. (IV.2) ($d_{ji} > 0$), the estimation error dynamics eq. (IV.1) can be rewritten as

$$\dot{\boldsymbol{e}}_j = W_\gamma(\boldsymbol{e}, \boldsymbol{x_o}, d) = W_0(\boldsymbol{e}_j, \boldsymbol{x_{o,j}}, d) + \gamma \Delta W(\boldsymbol{e}, \boldsymbol{x_o}, d), \tag{IV.3}$$

for some smooth function $\Delta W(\cdot,\cdot,\cdot)$ since the Kalman gain is linear with respect to $\boldsymbol{p}_j$. Note that if $(\boldsymbol{e}, d) = (0,0)$ then by definition of $\hat{\boldsymbol{v}}_{ji}$ in eq. (II.2) and $\boldsymbol{\mathcal{C}}_j$ in eq. (II.5) it holds $\boldsymbol{p}_j = \boldsymbol{x}_{\boldsymbol{o},j}^{GNSS} = \boldsymbol{x}_{\boldsymbol{o},j}$ or equivalently $W_0(0, \boldsymbol{x}_{\boldsymbol{o},j}, 0) = W_\gamma(0, \boldsymbol{x}_{\boldsymbol{o}}, 0)$ for all $\gamma \in [0,1]$ which implies that $\Delta W(0, \boldsymbol{x}_{\boldsymbol{o}}, 0) = 0$ for all $\boldsymbol{x}_{\boldsymbol{o}} \in \mathbb{R}^{6N}$. Moreover, if $d = 0$, $\bar{\boldsymbol{\chi}}_{ji}^-$ in eq. (II.3) is linear with respect to $\boldsymbol{e}$ and so is $\boldsymbol{p_j}$ (cfr. eq. (II.5)), then there exists $\beta > 0$ such that

$$||\gamma \Delta W(\boldsymbol{e}, \boldsymbol{x}_{\boldsymbol{o}}, 0)|| \leq \gamma\beta||\boldsymbol{e}||, \tag{IV.4}$$

for all $\boldsymbol{x}_{\boldsymbol{o}} \in \mathbb{R}^{N\cdot 6}$. Lemma 9.1 in [24] allows to conclude that $\boldsymbol{e} = 0$, with $d = 0$, is locally exponentially stable with sufficiently small selection of γ and sufficiently close to the origin $\boldsymbol{e} = 0$, i.e. for sufficiently small $\rho > 0$. To conclude, since the Kalman gain is linear with respect to d as it enters via $\boldsymbol{p_j}$ ($d_{ij} > 0$ by assumption), then the origin $\boldsymbol{e} = 0$ it is still exponentially stable for sufficiently small noise amplitude d such that $|d| \leq \bar{d}$. ■

Remark 1: Proposition 1 ensures the existence of a sufficiently small γ such that the proposed estimation algorithm locally (with respect to $\boldsymbol{x_o}$) exponentially converges if $\boldsymbol{p_j}$ is fed to the EKF, and the amplitude of the measurement noise d is sufficiently small. We recall that $\gamma \approx 0$ implies $\theta \approx 1$. The claim of proposition 1 ensure local exponential stability of the estimation error for all agents and mainly relies on the robustness of the EKF to sufficiently small perturbations in the estimation error dynamics introduced by $\boldsymbol{p_j}$. However, the proof does not provide a constructive procedure to select an opportune value for γ that has been instead selected by running heuristic optimisation on θ, and checking the algorithm convergence through Monte Carlo batch simulations.

Remark 2: The introduction of $\boldsymbol{p}_j$ creates interconnections among agents. By doing so the estimation error dynamics become fully coupled, whereas in the previous case ($\gamma = 0$) they were decoupled. In this paper, as indirectly stated by eq. (II.6), we have assumed that each agent can collect relative distances of agents within all fleet, i.e. we assume a complete graph topology with complete communication. However, also the cases in which some measurements are missing have been simulated numerically in section III-E.

V. Conclusions

The paper proposes a decentralized localization algorithm with a low computational cost, exploiting ranging measures between agents in a fleet of small-sats. We provided a comparative analysis of EKF and UKF by extensive simulations showing the higher effectiveness of the EKF. The position measurement combining GNSS and inter-satellite ranging measurements is defined in eq. (II.6). It allows to sensibly reduce the uncertainty of the estimated position if provided to the EKF, improving the precision by nearly one order of magnitude. A covariance analysis with extensive simulations is provided to support and show the effectiveness of the proposed solution. We also provide a stability analysis that guarantees the existence, although locally, of sufficiently low measurement disturbances allowing to tune the observer to ensure local exponential convergence of the estimation error. We recall that the decentralized solution assumes that some amounts of information are exchanged in real-time among the small-sats. In the case of large fleets, this might limit the proposed approach's applicability. However, simulations have revealed that the higher the number of satellites, the better the estimation performances are.

On the other hand, the communication model in a multi-agent system plays a crucial role in the algorithms. Therefore, we think a trade-off between the number of neighbors communicating their position and the transmission speed should be considered. Such a statement is supported by the results of section III-E, where we showed that even with a non-complete topology, the proposed algorithm performs better than a standalone GNSS-based Kalman filter. In the future development of this work, a similar approach to the one proposed for the position estimate could be used to improve the precision of the velocity estimation too. Moreover, it would be of great interest to investigate how the geometric disposition of the agents can affect the precision of the estimate obtained through ranging measures.

References

[1] *Starlink Project*. URL: https://www.starlink.com/ (cit. on p. 1).

[2] *Galileo Project*. URL: https://www.esa.int/Applications/Navigation/Galileo/Galileo_satellites (cit. on p. 1).

[3] Chee Khiang Pang et al. "Nano-satellite swarm for SAR applications: design and robust scheduling". In: *IEEE Transactions on Aerospace and Electronic Systems* (2015) (cit. on p. 1).

[4] Steven Engelen, Eberhard Gill, and Chris Verhoeven. "On the reliability, availability, and throughput of satellite swarms". In: *IEEE Transactions on Aerospace and Electronic Systems* (2014) (cit. on p. 1).

[5] *What is a CubeSat & other picosatellites*. URL: https://www.nanosats.eu/cubesat (cit. on pp. 1, 2).

[6] Alexander Chin et al. "CubeSat: The Pico-Satellite Standard for Research and Education". In: *AIAA SPACE 2008 Conference & Exposition* (2008) (cit. on p. 1).

[7] A. Toorian, K. Diaz, and S. Lee. "The CubeSat Approach to Space Access". In: *2008 IEEE Aerospace Conference* (2008) (cit. on p. 1).

[8] A. Giannitrapani et al. "Comparison of EKF and UKF for Spacecraft Localization via Angle Measurements". In: *IEEE Transactions on Aerospace and Electronic Systems* (2011) (cit. on p. 1).

[9] M. S. Challa, J. G. Moore, and D. J. Rogers. "A Simple Attitude Unscented Kalman Filter: Theory and Evaluation in a Magnetometer-Only Spacecraft Scenario". In: *IEEE Access* (2016) (cit. on pp. 1, 6, 8).

[10] M. D. Pham et al. “Gain-scheduled extended kalman filter for nanosatellite attitude determination system”. In: *IEEE Transactions on Aerospace and Electronic Systems* (2015) (cit. on p. 1).

[11] E.J. Lefferts, F.L. Markley, and M.D. Shuster. “Kalman Filtering for Spacecraft Attitude Estimation”. In: *Journal of Guidance, Control, and Dynamics* (1982) (cit. on pp. 1, 3).

[12] T. Uzay H.E. Soken. “Adaptive Unscented Kalman Filter for Small Satellite Attitude Estimation”. In: *Asia-Pacific International Symposium on Aerospace Technology* (2012) (cit. on p. 1).

[13] Vincent Giralo and Simone D’Amico. “Distributed multi-GNSS timing and localization for nanosatellites”. In: *NAVIGATION* (2019) (cit. on p. 1).

[14] Mahsa Mohammadi, Mahdi Baradarannia, and Ali Farzamnia. “Leader-following consensus of nonlinear multi-agent systems based on position and velocity estimations”. In: (2021) (cit. on p. 1).

[15] Bomin Jiang, Brian D. O. Anderson, and Hatem Hmam. “3-D Relative Localization of Mobile Systems Using Distance-Only Measurements via Semidefinite Optimization”. In: *IEEE Transactions on Aerospace and Electronic Systems* (2020) (cit. on p. 1).

[16] Mohammed Shalaby et al. “Relative Position Estimation in Multi-Agent Systems Using Attitude-Coupled Range Measurements”. In: *IEEE robotics & automation letters.* (2021) (cit. on pp. 1, 2).

[17] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2006 (cit. on pp. 1, 3).

[18] James R. Wertz. *Spacecraft Attitude Determination and Control*. Springer Netherlands, 1978 (cit. on pp. 1, 2).

[19] B. Wie. *Space vehicle dynamics and control*. AIAA Education Series, Reston, 1998 (cit. on pp. 1, 2).

[20] Marcelo J. Segura et al. “Ultra Wide-Band Localization and SLAM: A Comparative Study for Mobile Robot Navigation”. In: *Sensors* 11.2 (2011), pp. 2035–2055. ISSN: 1424-8220. DOI: 10.3390/s110202035. URL: https://www.mdpi.com/1424-8220/11/2/2035 (cit. on p. 2).

[21] Patrick Geneva, Kevin Eckenhoff, and Guoquan Huang. “Asynchronous multi-sensor fusion for 3D mapping and localization”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 5994–5999 (cit. on p. 2).

[22] Chi-Tsong Chen. *Linear system theory and design*. Saunders college publishing, 1984 (cit. on p. 2).

[23] Gerhard Haßlinger and Oliver Hohlfeld. “The Gilbert-Elliott Model for Packet Loss in Real Time Services on the Internet”. In: *MMB*. 2008 (cit. on p. 3).

[24] Hassan K Khalil. *Nonlinear systems; 3rd ed.* Prentice-Hall, 2002 (cit. on p. 11).