Infrastructure-free Multidimensional Scaling for drones swarm localization

Giacomo Mutti, *233189*,
Pierfrancesco Oselin, *229564*,
Riccardo Periotto, *230266*

***Abstract*— This paper explores the localization of drones within various operational scenarios using the Multidimensional Scaling (MDS) algorithm and compares its performance with the conventional trilateration method. Drones' widespread applications, ranging from aerial photography to scientific research, necessitate precise localization even in challenging environments where GPS signals are unreliable. The MDS algorithm, capable of leveraging Ultra-Wideband (UWB) measurements, is investigated for its accuracy and robustness in comparison to trilateration. Through numerical and Software-in-the-loop simulations, MDS consistently outperforms trilateration across scenarios involving Gaussian-distributed noise in dynamics, measurements, and clock-time. Notably, this study pioneers swarm estimation via infrastructure-free measurements, wherein a designated anchor node collects data to drive the algorithm. Future research avenues are suggested, including extended-MDS integration and covariance estimation for enhanced Kalmann filter updates. This work contributes insights into optimal drone localization methods for diverse applications, highlighting MDS as a superior choice for precision and adaptability.**

I. Introduction

DRONES have witnessed escalating utilization across diverse sectors, revolutionizing industries with their aerial capabilities. From aerial photography, surveillance, search and rescue to agriculture, infrastructure inspection, and delivery services, drones are reshaping operational paradigms. Their ability to navigate challenging environments, gather real-time data and efficiently execute tasks has led to their integration in sectors such as filmmaking, disaster response, precision farming, and scientific research. As technological advancements continue, drones' versatile applications are expected to further expand, driving innovation and transforming industries on a global scale.

Regardless of the application, the ability to determine the exact location of a drone is critical to achieving operational goals. For example, precise localization boosts safe navigation, optimal data collection, and effective communication. More generally, it is possible to claim that it increases the chances of successful drone adoption. Drones are typically localized using Global Positioning Systems (GPS), such as Global Navigation Satellite System (GNSS), which rely on satellites' signals to determine their position. In addition, sensors such as accelerometers, gyroscopes, and barometers aid navigation and stabilization. However, this global reference is not always available, especially as the range of applications stretches to remote and hardly reachable locations: indoor settings, urban canyons, and dense vegetation. This poses a challenge to which the answer is still a topic of research.

A. Possible solutions

In these circumstances, some techniques like visual odometry and SLAM (Simultaneous Localization and Mapping) are employed for computing the drone position and attitude. These methods enable drones to navigate autonomously, avoid obstacles, and execute tasks with precision. An alternative approach, more suitable for a swarm of drones, is represented by Ultra-Wideband (UWB) signals. UWB technology operates by transmitting short-duration pulses with extremely low power across a wide frequency spectrum. UWB signals experience minimal interference and exhibit high immunity to multipath effects, making them ideal for precise distance measurement. This enables UWB-equipped drones to establish relative distances between themselves.

This set of relative distances represents a snapshot of the swarm configuration and can be used to compute the relative layout through various algorithms. Among these, trilateration plays the role of the most straightforward one. This method derives the position of the nodes considering their distances from 3 known points in the 2D case and 4 points in the 3D one (see Section III for more details). The problem can be solved with different techniques depending on the settings. If the measurements are not disturbed, the problem becomes an algebraic system of equations. On the other hand, if data are corrupted, Least-Square Minimization (LSM) can be used. Although valuable, a shortcoming of the trilateration approach is that it only uses the distances between the nodes and the anchors, not taking advantage of the information contained in the distance measurements between non-anchor nodes.

The Multidimensional Scaling (MDS) algorithm is a favored approach for leveraging UWB measurements in wireless network localization. MDS is a widely employed method for dimensionality reduction, which objective is to transforms multidimensional data into a lower-dimensional space while retaining essential information. While various dimensionality reduction techniques like PCA, factor analysis, and isomap exist, MDS stands out due to its user-friendly nature and versatile applicability.

B. Contribution

The objective of this paper is to compare the performance of the two mentioned localization algorithms, namely trilateration

through LSM and MDS, within a ROS 2[1] project. Through testing scripts that utilize these algorithms, the superior qualities of MDS become evident when contrasted with trilateration. MDS demonstrates robustness and precision, particularly in scenarios characterized by noise and measurement errors. The project's code can be found at the repository below [2].

The rest of the paper is organized as follows. Section II delves into the related work, presenting the currently employed Local Positioning System (LPS) techniques.

Subsequently, Section III provides a comprehensive elucidation of the mathematical principles behind both algorithms. In Section IV, the project's structure is described, with an explanation about the algorithms execution, how it has been developed the simulation suites and the experiments that has been conducted.

Finally, results and conclusion collect the outcomes of the study and present a summary of what was done, mentioning some possible future directions.

II. Related Work

The problem of determining the position of the nodes in a Wireless Sensor Network (WSN) has been a subject of research since the inception of these networks [1], [2]. Initially, the focus was on tracking sensor nodes within the network, as their data was inherently valuable only when coupled with accurate positional information. Over time, the convergence of WSNs with the Internet of Things (IoT) and the spread adoption of unmanned vehicles have sustained the interest in this field with application in both 2D and 3D scenarios [3].

While GPS have conventionally been employed to address localization challenges, its utilization is limited by factors such as the unavailability of global measurements in certain environments and high costs in terms of size, price, and energy consumption [1]. Consequently, a trend followed in the literature to overcome these limitations has been the adoption of Local Positioning Systems (LPS).

In the field of LPS, a notable approach is utilizing radio frequency (RF) signal propagation for accurate localization. Ultra-Wideband (UWB) technology, known for its robustness against multipath errors, obstacle penetration, high accuracy, and cost-effectiveness, has garnered increasing interest [4], [5], [6]. Common techniques for RF-based localization include Received Signal Strength (RSS), Time of Arrival (ToA), Time Difference of Arrival (TDoA), and Angle of Arrival (AoA) methods. Despite the differences between these methods, the ultimate goal is to provide a measure of the physical distance between sensors.

Recent studies have explored UWB-based localization approaches, with a focus on static infrastructure utilizing fixed nodes (anchors) to locate other nodes (agents) within their range [7], [8]. In 3D cases, when the distance of an agent from 4 anchors is known, various techniques have been developed to estimate node locations based on distance measurements [9, Chapters 23-27]. Among the others, as already mentioned in Section I, trilateration was implemented and tested through LSM, due to its simplicity and efficacy. Despite these properties, this work aims at demonstrating that the trilateration method does not fully exploit the available information when all the nodes are equipped with UWB sensors. The reason behind this claim is that, with this setting, all the relative distances between nodes are known, not just those between the agents and the anchors, suggesting the potential for improved performance.

Among the algorithms that fully leverage all the relative measures, MDS and its derivatives emerge. Rooted in psychology and psychometrics [10], [11], [12], the ability of MDS to map high-dimensional information to lower dimensions has found applications across diverse fields.

When used for localization, MDS is usually referred to as metric MDS, to underline that the distances correspond to Euclidean distance between nodes [13]. In [14], the authors demonstrate that the classical MDS algorithm is feasible for mobile localization and that the corresponding estimator does not rely on the initial estimate of the anchors. In the same work, they point out that a common weakness of standard MDS methods is that the solutions do not consider the range noise. However, the exploration of weighted MDS approaches has addressed this limitation by incorporating measurement error covariance in the formulation and improving the position estimates [14]. This formulation, although more complex, demonstrated optimal performance according to the Cramèr-Rao lower bound limit (variance-optimality) [14].

In a comprehensive review of outdoor and indoor positioning systems, recent work has detailed various MDS variants, including centralized, semi-centralized, and distributed versions [13]. In this research, the effort was focused on the centralized version of MDS, focusing on its application to drones. A work similar to this is the infrastructure-free one proposed in [15], in which the swarm mapping occurs relative to few drones designated as moving "anchors" and not with respect to fixed anchors preinstalled in the mission environment.

Inspired by this trend, the paper addresses the limitations of static infrastructure by proposing an innovative approach where a single drone serves as the UWB anchor for the entire swarm, as it will be better described in Section IV. Essentially, after calculating a relative map by considering the distances between all the drones, ambiguities are resolved by considering the last four sets of measurements. After this, the anchor is moved to a new point from which a new set of measurements and estimation are performed (see Section IV).

To conclude, other recent approaches are those based on multi-lateration, where the idea behind trilateration is applied several times. The advantage of this approach is that it is not necessary to know the distance between each pair of drones, as long as the known distances are enough to calculate the position of new drones from the known ones. However, MDS provides a more elegant and efficient formulation of the problem, simultaneously computing the position of all nodes [2].

[1] Robot Operating System v2

[2] https://github.com/oselin/drone-pose-estimation

III. Theory

In order to solve the problems introduced in Section I, effective methods have to be exploited to provide reliable results. Particularly, this paper focused on pursuing the research quest with one specific algorithm, known in the literature as Multidimensional Scaling (MDS) algorithm.

The algorithm, as will be later explained, uses the knowledge of relative distances among the set of points, or set of UAVs in this case, to correctly estimate the pose in space and obtain the correspondent Cartesian coordinates. The approach, however, suffers from ambiguity problems that can be solved only if a minimum of $n+1$ measurements are taken, with n dimension of the search space. Since the same amount of information is needed for computing the classical trilateration algorithm, this will be directly compared to MDS to highlight what might be the benefits or downside for introducing a more complex but robust algorithm.

A. Trilateration algorithm

Trilateration is a geometric methodology that allows to determine the precise spatial coordinates of an unknown point within a given dimensional space, typically 2D or 3D, through the utilization of distance measurements from predetermined reference points. Particularly, the method relies on $n+1$ reference points, known in the literature as anchors, with n dimension of the search space.

In details, this algorithm introduces circumferential boundaries, represented as spheres in 3D or circles in 2D, centered at each reference point. The respective radius corresponds to the measured distances. The point of interest is subsequently localized as the intersection juncture of these circumferences or spheres.

Given X_i the unknown coordinates of the i-th node to be located and, Q_j the coordinates of the j-th anchor and $d_{i,j}$ the distance between X_i and Q_j, the trilateration problem can be solved via LSM by minimizing the loss function reported in Equation 1.

$$\hat{X}_i = \min_{X_i} \sum_{j=1}^{n} (||X_i - Q_j|| - d_{i,j})^2 \tag{1}$$

Even though the algorithm is simple, fast and effective, its performances decrease rapidly if incorrect information are introduced, as there is no validation nor correction of noisy measurements. Additionally, if the anchors are absent or the positions of the anchors are not available, the algorithm cannot be applied [16].

Figure 1 provides a visual representation of the just discussed algorithm: it is possible to observe 4 different spheres spotting the unknown point. The respective radii are the measured distances.

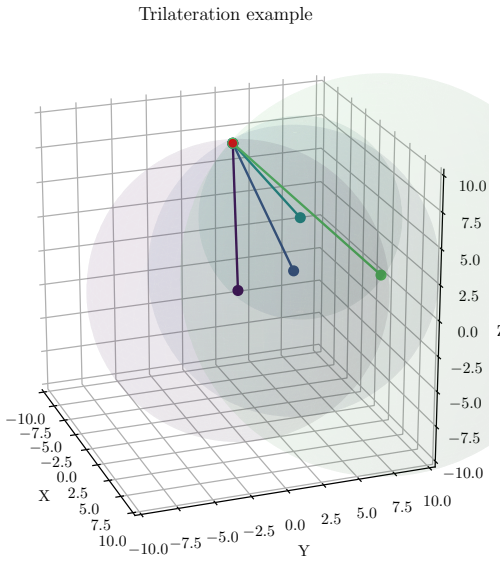


Fig. 1: **Trilateration example.** 4 spheres detecting an unknown point in space, respecting the $n+1$ rule. The targeted point is found as the intersection of spheres, with no ambiguities.

B. Multidimensional Scaling algorithm

MDS is a robust methods that allows to estimate points coordinates given the relative distances among them. Since it relies on fully-connected points, the algorithm results to be robust against noise, as one information, i.e. distance, is validated by another one. Indeed, given two generic points X_i and X_j, the distance information $d_{i,j}$ obtained from the i-th point is supported by the distance $d_{j,i}$ read from the j-th point. In this way, the method is less dependent on the single piece of information that might be inaccurate, leading to a more balanced solution. Figure 2 shows an example of fully-connected network of points, in which each node exchanges distance information with the others.

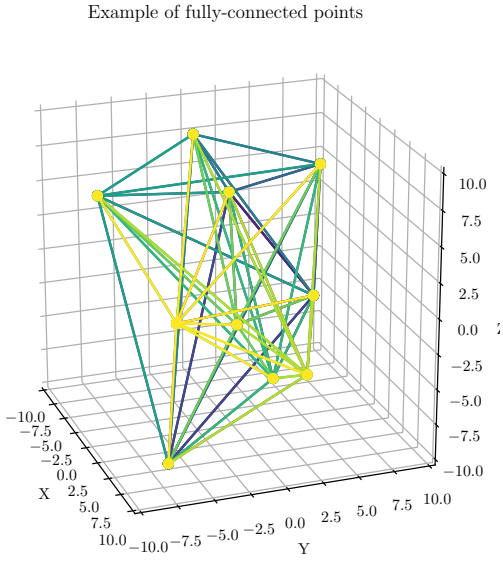


Fig. 2: **Example of fully-connected network.** Each point communicates with the others, exchanging distance information. The Latin Hypecube Sampling was used for maximizing the distribution of points for this representation. The represented scenario is the input for the MDS algorithm.

The algorithm can be exploited for several applications, but a natural application lies in the estimation of the location of points in space. Specifically, it uses a square matrix of squared distances to build a relative map that describes the distribution of points in space [16]. An example of distance matrix is reported in Equation 2. In details, $d_{i,j}$ represents the squared distance between point i and point j. Naturally, $d_{j,i}$ represents the squared distance between point j and point i. In an error- and noise-free scenario, $d_{i,j} = d_{j,i}$ whilst in a real-case application the two elements might differ due to background noise, latency or interference.

$$D = \begin{bmatrix} d_{1,1} & d_{1,2} & \dots & d_{1,N} \\ d_{2,1} & d_{2,2} & \dots & d_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ d_{N,1} & d_{N,2} & \dots & d_{N,N} \end{bmatrix} \tag{2}$$

The algorithm results to be easy and flexible and can be summarized in two major steps: *1)* estimation of a relative map from the distance matrix and *2)* transformation of the relative map in the absolute one. The following paragraphs better describe the two steps.

1) From distance matrix to relative map: The core of MDS relies on the eigendecomposition of a symmetric matrix D of square distances to estimate the relative map of the involved points.

Given the D matrix, it is possible to write the associated eigendecomposition by introducing the centering operator $\mathbf{H} = \mathbf{I} - \mathbf{e}\mathbf{e}^T/N$, with N dimension of the matrix. The new centered matrix is reported in the left-hand side of Equation 3, while the associated decomposition on its right-hand side.

$$-\frac{1}{2}\mathbf{HDH} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \tag{3}$$

Eventually, the relative map $\tilde{X}$ can be estimated by computing

$$\tilde{X} = \mathbf{\Lambda}^{\frac{1}{2}}\mathbf{U}^T \tag{4}$$

The relative map is a map that describes the location of points in space. It might be affected by unknown Cartesian transformation such as translation, rotation and flip. The fact that a relative configuration can correspond to multiple absolute ones has been called in the literature ambiguity.

2) From relative to global map: In order to turn a relative map to a global map, i.e. the correct estimation of the involved nodes, the ambiguities introduced in Section III-B1 must be completely solved. In order to achieve that, at least $n+1$ points with known location have to be used.

Several methods are known in the literature to solve such problem. Most of them are explained in details in [17]. However, thanks to its reliability and accuracy, the Singular Value Decomposition (SVD) method has been chosen for this application, and the following paragraph will explain it in details.

Let $p = [p_1, p_2, \dots, p_{n+1}]$ and $q = [q_1, q_2, \dots, q_{n+1}]$ two sets of corresponding anchors, where q represents the set of true coordinates and p the estimated one coming out from the eigendecomposition step.

1) Compute the weighted centroids of both point sets

$$\bar{p} = \frac{1}{n+1}\sum_{i=1}^{n+1} p_i, \quad \bar{q} = \frac{1}{n+1}\sum_{i=1}^{n+1} q_i \tag{5}$$

2) Compute the centered vectors

$$p'_i = p_i - \bar{p}, \quad q'_i = q_i - \bar{q}, \quad \forall i \in [1, n+1] \tag{6}$$

3) Compute the $n \times n$ covariance matrix

$$C = P'Q'^T \tag{7}$$

where P' and Q' are the $n \times (n+1)$ matrices that have p'_i and q'_i as their columns, respectively.

4) Compute the singular value decomposition

$$C = U\Sigma V^T \tag{8}$$

The unknown rotation can be obtained as

$$R = VU^T \tag{9}$$

while the unknown translation as

$$t = \bar{q} - R\bar{p} \tag{10}$$

Therefore, the global map can be written as

$$\hat{X} = R\tilde{X} + t \tag{11}$$

Please note that with the SVD method, no flip ambiguities must be solved.

IV. Methodology

This section describes the main aspects of the implementation not yet adequately depicted and the methodologies employed to verify the performances of the discussed techniques. The implementation consists of a ROS 2 project where the goal is to estimate the position of the drones in a swarm using the measurements of their relative distances. The swarm operates in a simulated environment, which can be of two types. In the first type, a dedicated ROS 2 node is responsible for simulating the drones dynamics solely via numerical integration, while the second adopts the physical simulator Gazebo[3]. To make this second simulation even more realistic, the commands are routed through ArduCopter[4] via Mavros[5].

The reason why two simulation types were developed is twofold. First, in this way the project is more modular and scalable. Second, running a simulation with Gazebo and a Mavros node for each drone is computational heavy and no adequate computational resources were available for this purpose. Still, the Gazebo+Ardupilot combination serves the purpose of closely simulating the Software-In-The-Loop (SITL) environment, bridging the gap towards real-world deployment. Figure 3 illustrates the Gazebo setup in the case of 3 drones.

In the following, the process regarding the data collection for such scenario will be better explained. Subsequently, the blocks composing the ROS 2 project will be better illustrated, highlighting how they interact, both in the case of simple numerical simulator and with Gazebo. Finally, the experiment procedure will be exposed and explained, leading to the comparison of the two algorithms, i.e. trilateration and MDS.

[3]https://gazebosim.org/home
[4]https://ardupilot.org/copter/
[5]https://github.com/mavlink/mavros

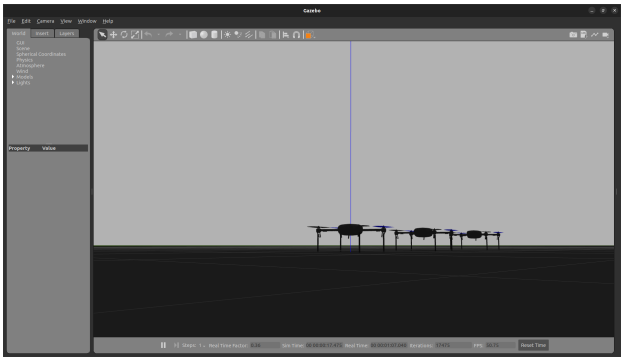


Fig. 3: **Gazebo setup.** Gazebo simulator and runway world for simulating the guidance of a swarm of drones while estimating their positions.

A. *Infrastructure-free localization*

In this section, the approach for an infrastructure-free localization of drones is presented, operating within a WSN. Unlike traditional methods that rely on a fixed set of anchor nodes to determine drone locations, this research embraces a novel approach inspired by recent trends in the literature. This approach involves designating a single drone within the swarm as the "anchor node", which moves faster than the others and is in charge of collecting distance measurements from various points. As reiterated multiple times, both of the compared algorithms are only able to solve the problem if equipped with 4 sets of measurements from distinct points.

In the project, the reference system with respect to which the positions of the drones are calculated is centered at the position of the anchor. In both the simulation types, the drone designated as the anchor is the one called *drone1*. In the case of the numerical simulation, its initial position is $x_0^1 = [0.0, 0.0, 5.0]^\top$[6], while in Gazebo it is in at the origin of the absolute reference frame. In contrast, the position of the other drones is randomly generated in all coordinates.

During the simulation, the swarm is moved at a constant speed in a predefined direction. The anchor drone is subject to the same movement of the swarm, but, in addition to it, it also continuously changes its position to collect new measurements from different points. This is achieved by imposing to the anchor not only the swarm velocity, but also an additional one depending on the direction it has to follow.

Once the anchor has moved in a certain direction for a given time (which can be affected by noise), it remains still relatively to the swarm, to complete the distances collection and perform an estimation of the swarm configuration through the localization algorithms. After this, a new direction is assigned to the anchor and the cycle is repeated. Figure 4 presents a flowchart with the main steps of the described approach, while Figure 5 the screenshots obtained by executing the project with no noise for an anchor movement.

[6]In the notation, subscript refers to time, and superscript to drone.

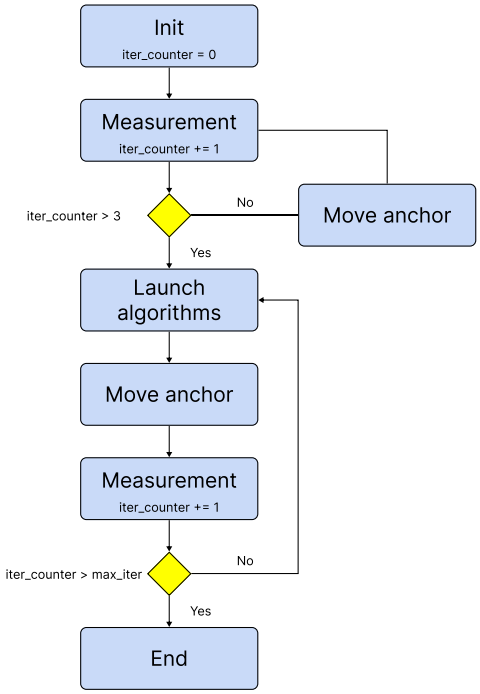


Fig. 4: **Software flowchart.** The diagram illustrates the main steps of the implemented approach. This was used for both the simulation types. The first loop is necessary to make sure to have at least four sets of measurements. The second cycle, on the other hand, is the one continuously performed to estimate the drones position.

B. *Architecture*

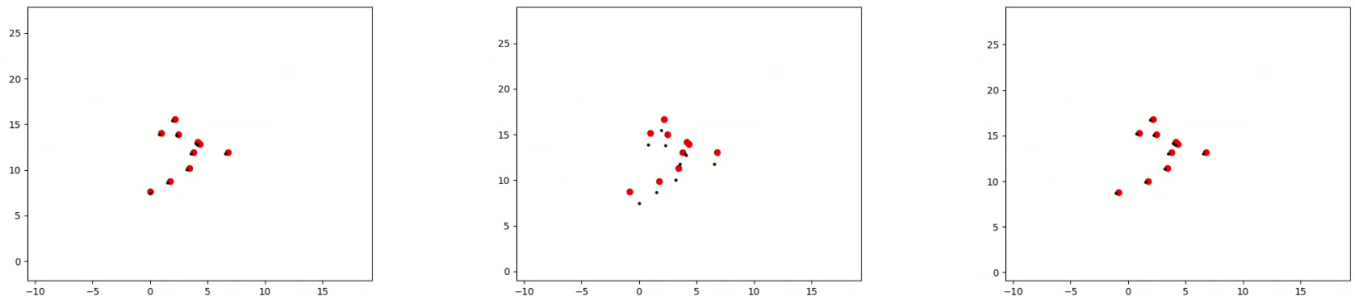
The project is built upon ROS 2, with key nodes designed to fulfill specific roles within the system architecture. The core node is the one called *main*, which is responsible for guiding drones, reading distance measurements, running algorithms and tracking results. Another important node, called *hub*, reads the coordinates from the environment (either the numerical simulator or Gazebo) and returns the distances, possibly adding noise. This functionality simulates the adoption of real UWB sensors.

The numerical simulator node (i.e. *test*) is responsible for tracking drones' positions, simulating the application of velocity commands, and incorporating dynamic model noise to represent real-world motion behaviors. Other nodes were adopted when using Gazebo: the Mavros nodes, for communicating between the scripts, and the Ardupilot plugin.

The simplified diagram in Figure 6 provides a visual representation of the simulation setup, highlighting the main nodes and topics[7]. The diagram resembles the one obtained by launching the rqt_graph[8] tool. For clarity, only the connections of 3 drones are illustrated.

[7]In ROS 2, a topic is a named channel of communication that facilitates the exchange of data between different nodes.

[8]http://wiki.ros.org/rqt_graph



(a) Step 1. (b) Step 2. (c) Step 3.

Fig. 5: **Algorithm iteration.** This screenshots illustrate the main steps of an iteration of the algorithm. The anchor starts moving from a point in which the data collection and estimation have been performed (5a). While moving, the rest of the swarm moves toward a predefined direction, positive y in the case shown (5b). After a given time, the anchor stops and remains in the same relative position for a new data collection and estimation, until the next iteration (5c).

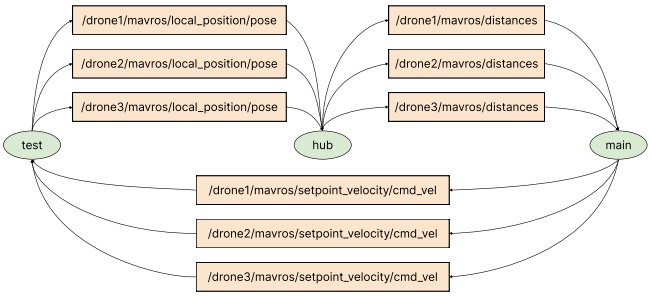


Fig. 6: **Nodes architecture.** Graph showing the connections between the main nodes operating when running the project for 3 drones with the numerical simulator. Each node is represented with an ellipse, while the topics are in rectangles. The graph has bee redrawn from that obtained using the rqt_graph tool. Some of the connections needed in the real implementation have been deleted for clarity.

C. Experimental setup

To evaluate the performance of the two algorithms, a series of targeted experiments was carried out, mostly using the numerical simulator. This approach enabled to run multiple experiments with different settings and aggregate performance metrics for robust conclusions. These experiments encompassed distinct settings, each characterized by specific noises. The types of noise considered are: a) the noise affecting the distance measurement, which is set by the hub; b) the noise that affects the time for which a node continues to move along a specific direction, compromising the accurate knowledge of the anchor position; c) a noise affecting the system dynamics.

All the noises are assumed to come from Gaussian distributions centered in zero and with a standard deviation specified via parameters.

The algorithms were assessed across five distinct settings: no noise, one noise out of three per time, and all three present together. Table I summarizes the standard deviations characterizing the different noises for the tested setting.

Each setting underwent 50 runs with 33 anchor movements and 30 algorithm executions (i.e. 30 iterations of the second cycle in the diagram in Figure 4 started after having collected 3 set of measurement). Each time the project is executed, the number of drones can be specified by a parameter. For data collection, the control of 10 drones was simulated, whose location is randomly generated using the number of runs being tested as a seed[9], so that the comparison between different settings is fair. In addition, to facilitate the comparison and make it more meaningful, average errors are calculated by grouping both by setting and drone, as shown in the following section.

Quantity	**setting1**	**setting2**	**setting3**	**setting4**	**setting5**
Distance [m]	0.0	0.05	0.0	0.0	0.05
Time [s]	0.0	0.0	0.0	0.2	0.2
Dynamics [m]	0.0	0.0	0.001	0.0	0.001

TABLE I: **Setting details.** The standard deviations of the Gaussian-distributed, zero-mean noises affecting the considered quantities is reported for the different tested settings.

V. Results

This section presents the outcomes obtained through the implementation and execution of the trilateration and MDS algorithms for estimating the positions of drones in the scenarios and with the settings previously presented.

Table II summarizes the main focus of this study: a comparative evaluation of the estimation errors achieved by employing the two algorithms. As expected, these findings affirm the superiority of MDS in terms of average estimation errors.

To compute the average performances, point-wise errors were calculated for each drone at every timestep, and grouped the results by settings. This categorization provides insights to the consistent outperformance of MDS over trilateration, regardless of the nature of the simulated noise impacting the execution.

Another result regards the value of the average estimation error. For each drone,it continuously increases iteration by iteration.

[9]In the context of a random number generator (RNG), a seed is an initial value that is used to start the generation process. It serves as the starting point for producing a sequence of seemingly random numbers.

Algorithm	setting1	setting2	setting3	setting4	setting5
MDS	0.21	0.31	0.25	0.55	0.57
Trilateration	0.35	0.41	0.34	0.83	0.86

TABLE II: **Average estimation errors by tested settings.**

This trend, observable in Figure 7, can be attributed to the influence of the drift, which is mostly caused by the action of the noises on the system. Additionally, it can be also related to the discrete-time nature of these computations, and it might be influenced by the timestep used for executing the project nodes.

The comparison of different noises, based on the data collected, presents challenges due to varying attributes and quantity involved (e.g. distances, times, positions), but their effects are evident, especially for Setting 4 and Setting 5, observable respectively in Figure 7d and Figure 7e.

This outcome underlines the potential for enhancements in error mitigation strategies against noises and drifts, such as the introduction of global measurements or absolute re-calibration procedures, as elaborated in the last section.

Table III illustrates the great performances of MDS over trilateration. Indeed, the algorithm provided an error reduction between 24.4% and 40.0%. A noticeable information is that an error reduction of about 33.7% was achieved in Setting 5, namely the most noisy and complex scenario tested. This directly highlights the strong capabilities and reliability of MDS.

However, few improvements could be introduced to obtain even higher results. First of all, different MDS versions could be integrated, to better overcome the different type of noise and reduce the computational times. Example of those are enhanced Multidimensional Scaling (eMDS) [18], dynamic Multidimensional Scaling (dMDS) [19], MDS-MAP or GM-MDS.

Configuration	Error reduction
Setting1	40.0%
Setting2	24.4%
Setting3	26.5%
Setting4	33.7%
Setting5	33.7%

TABLE III: **Average error reduction by relying on MDS algorithm**: in the table the reduction in the average error is shown, by relying on the MDS algorithm for drones coordinates estimation instead of standard trilateration algorithm.

Additionally, partially-connected swarms could be studied and implemented with the introduction of distributed MDS and distributed-weighted MDS (dwMDS), to better face the weakness or absence of intercommunication among the drones, due to the presence of obstacles.

Another interesting research is the integration of such simulation with Kalmann filters, to better estimate the position of drones via a measurement-update approach. In particular, the estimation of the covariance matrix associated to the location of each drone could be used to update more accurately the covariance in the filter, in order to take advantage of the best features of both the tools.

VI. Conclusion

This paper illustrated the implementation of the Multidimensional Scaling algorithm and the performances that can be obtained when it is used for estimating the location of a swarm of drones. Particularly, two different simulations were assessed to demonstrate its potential: a numerical one to collect data and to compute the average behaviour and the associated error, and a Software-in-the-loop one to demonstrate potential real-world applications.

The performance metrics were compared to a more classical but less robust algorithms known in the literature as trilateration. It has demonstrated that MDS always outperformed trilateration in all the different studied scenarios, in which different Gaussian-distributed noise was added to dynamics, measurement or clock-time.

The method studied in this work introduces for the first time the estimation of a swarm of fully-connected drones via infrastructure-free measurements, in which one node, designated as anchor, is responsible to move around and collect different measurements to make the algorithm run.

Future improvements and further studies can be conducted, such as the introduction of an extended-MDS or the integration of its covariance estimation to better update a Kalmann filter.

References

[1] L. Doherty, K. Pister, and L. El Ghaoui, "Convex position estimation in wireless sensor networks," in *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*, vol. 3. IEEE, pp. 1655–1663. [Online]. Available: http://ieeexplore.ieee.org/document/916662/

[2] P. Corbalán, G. P. Picco, M. Coors, and V. Jain, "Self-localization of ultra-wideband anchors: From theory to practice," vol. 11, pp. 29 711–29 725. [Online]. Available: https://ieeexplore.ieee.org/document/10080967/

[3] Popescu, Stoican, Stamatescu, Chenaru, and Ichim, "A survey of collaborative UAV–WSN systems for efficient monitoring," vol. 19, no. 21, p. 4690. [Online]. Available: https://www.mdpi.com/1424-8220/19/21/4690

[4] L. Santoro, M. Nardello, D. Brunelli, and D. Fontanelli, "UWB-based indoor positioning system with infinite scalability," vol. 72, pp. 1–11. [Online]. Available: https://ieeexplore.ieee.org/document/10143252/

[5] V. Niculescu, D. Palossi, M. Magno, and L. Benini, "Energy-efficient, precise UWB-based 3-d localization of sensor nodes with a nano-UAV," vol. 10, no. 7, pp. 5760–5777. [Online]. Available: https://ieeexplore.ieee.org/document/9756977/

[6] J. P. Queralta, C. M. Almansa, F. Schiano, D. Floreano, and T. Westerlund, "UWB-based system for UAV localization in GNSS-denied environments: Characterization and dataset," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4521–4528. [Online]. Available: http://arxiv.org/abs/2003.04380

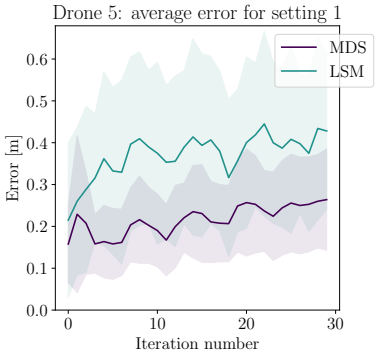
[7] C. M. Almansa, W. Shule, J. P. Queralta, and T. Westerlund, "Autocalibration of a mobile UWB localization system for ad-hoc multi-robot deployments in GNSS-denied environments." [Online]. Available: https://arxiv.org/abs/2004.06762

[8] N. Patwari, J. Ash, S. Kyperountas, A. Hero, R. Moses, and N. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," vol. 22, no. 4, pp. 54–69. [Online]. Available: http://ieeexplore.ieee.org/document/1458287/

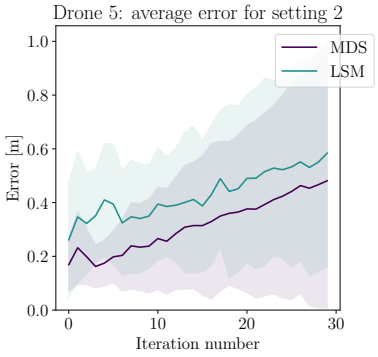
[9] S. A. Zekavat and M. Buehrer, *Handbook of position location: theory, practice and advances*, ser. IEEE series on digital & mobile communication. Wiley-Blackwell.

[10] S. L. France and J. D. Carroll, "Two-way multidimensional scaling: A review," vol. 41, no. 5, pp. 644–661. [Online]. Available: http://ieeexplore.ieee.org/document/5613204/

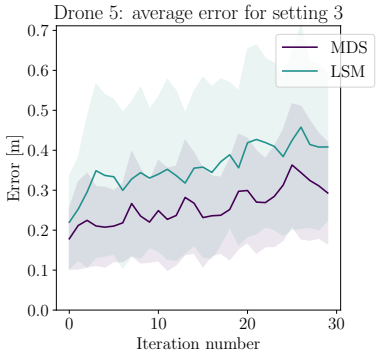
[11] G. Young and A. S. Householder, "Discussion of a set of points in terms of their mutual distances," vol. 3, no. 1, pp. 19–22. [Online]. Available: http://link.springer.com/10.1007/BF02287916



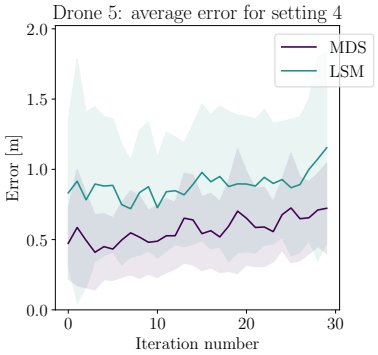
(a) Setting 1.



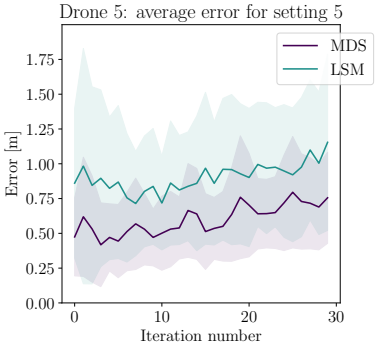
(b) Setting 2.



(c) Setting 3.



(d) Setting 4.



(e) Setting 5.

Fig. 7: **Drift.** The five graphs illustrate the performances over time for the two considered algorithms. Specifically, the average estimation error and its trend for drone5 (chosen arbitrarily) is shown for the different tested settings. Independently, the average error always increases, highlighting the drift phenomenon.

[12] W. S. Torgerson, "Multidimensional scaling: I. theory and method," vol. 17, no. 4, pp. 401–419. [Online]. Available: http://link.springer.com/10.1007/BF02288916

[13] N. Saeed, H. Nam, T. Y. Al-Naffouri, and M.-S. Alouini, "A state-of-the-art survey on multidimensional scaling-based localization techniques," vol. 21, no. 4, pp. 3565–3583. [Online]. Available: https://ieeexplore.ieee.org/document/8734111/

[14] Zhang-Xin Chen, He-Wen Wei, Qun Wan, Shang-Fu Ye, and Wan-Lin Yang, "A supplement to multidimensional scaling framework for mobile location: A unified view," vol. 57, no. 5, pp. 2030–2034. [Online]. Available: http://ieeexplore.ieee.org/document/4760257/

[15] M. Pourjabar, A. AlKatheeri, M. Rusci, A. Barcis, V. Niculescu, E. Ferrante, D. Palossi, and L. Benini, "Land & localize: An infrastructure-free and scalable nano-drones swarm with UWB-based localization." [Online]. Available: http://arxiv.org/abs/2307.10255

[16] W. Li, B. Jelfs, A. Kealy, X. Wang, and B. Moran, "Cooperative localization using distance measurements for mobile nodes," *Sensors*, vol. 21, no. 4, 2021.

[17] B. Risteska Stojkoska, "Nodes Localization in 3D Wireless Sensor Networks Based on Multidimensional Scaling Algorithm," *International Scholarly Research Notices*, vol. 2014, pp. 1–10, 10 2014.

[18] C. Di Franco, A. Melani, and M. Marinoni, "Solving ambiguities in mds relative localization," 07 2015.

[19] K. Ambrosi and J. Hansohm, "Ein dynamischer ansatz zur repräsentation von objekten," in *DGOR*, H. Isermann, G. Merle, U. Rieder, R. Schmidt, and L. Streitferdt, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 425–431.