Distributed Gradient Seeking

Paolo Furia *paolo.furia@studenti.unitn.it* ID 239451
Lorenzo Colturato *lorenzo.colturato@studenti.unitn.it* ID 233301

***Abstract*—This project explores the design and implementation of a distributed system for autonomous navigation, where a ground vehicle aims to move from an initial location to a destination point at the center of a scalar field following the gradient of the field. The ground vehicle is equipped with an Extended Kalman Filter (EKF) to estimate its position because it has no direct knowledge of its true position. The measurement model consists of estimates of its true position derived by a network of drones via Time-Difference-Of-Arrival (TDOA) measurements. By receiving the signal emitted by the ground vehicle at each time instant and by collecting information from its neighbors, each drone is able to compute locally the TDOA measurements and, using them in its own EKF, it estimates the position of the vehicle. To ensure a robust and accurate estimation, the project introduces a weighted average consensus algorithm. This algorithm gives more weight to drones with a larger number of measurements, thus improving the reliability of the collective estimates. The consensus process propagates these weighted estimates through the drone network, eventually leading to a global agreement on the position of the ground vehicle. This overall estimate is then communicated to the ground vehicle, which integrates it into its EKF for continuous position updating and navigation. The distributed nature of the system, combined with the consensus algorithm, ensures that the ground vehicle can reliably and precisely navigate to its target point, even in the presence of uncertainties and without direct knowledge of the position.**

Acronyms

EKF	Extended Kalman Filter.
GPS	Global Positioning System.
PID	Proportional–Integral–Derivative.
TDOA	Time-Difference-Of-Arrival.
TOA	Time-Of-Arrival.

I. Introduction

The task of guiding a unicycle robot to the extrema of an unidentified scalar ambient field is examined in this study. Finding and locating the sources of hazardous chemical leaks or vapour emissions, sources of pollutants and plumes, hydrothermal vents, etc. are a few examples of missions where steering to extrema is of interest. This type of problem is commonly referred to in the literature as source/location search problems.

The unicycle robot navigate within an area containing a partially unknown scalar field $D(\boldsymbol{r})$ where $\boldsymbol{r} = (x, y)$. The objective is to direct the robot to a point designated as $\boldsymbol{r}^0$, where the field distribution $D(\boldsymbol{r})$ exhibits its maximum value. The robot is equipped with a sensor capable of measuring only the field value $d = D(x, y)$ at the robot's current location (x, y) and the rate $\dot{d}$ at which this measurement evolves over time. In particular, the entire gradient $\nabla D(x, y)$ is not accessible. More attention has been given to the estimation of the robot's state. Multiple mobile drones are equipped with a sensor capable to receive the signal emitted periodically by the moving robot and therefore they act as receivers. If multiple receivers detect the same signal, it is possible to infer the robots's location using the detection times at these receivers. Each drone operates locally by exchanging information only with its neighbors. In particular, the Time-Of-Arrival (TOA) of the emitted signal is exchanged between neighboring drones and each drone computes locally the TDOA measurements, by subtracting its detection time with the detection time of all its neighbors. Assuming that the propagation speed of the signal is known, the TDOA measurements can be converted to range-difference measurements, which can then be used to estimate the location of the target by means of an EKF, which is run locally by each drone. In this decentralized approach each drone assumes itself as the reference drone, and generates TDOA measurements from its neighbors. Ennasr et al. [2016]

Although the robot moves on the ground and therefore its movement remains confined to the xy-plane, the drones, being in the air, are influenced by their altitudes and consequently perform a 3D estimate. Given the distributed nature of the proposed system of drones, in order to estimate the robot's position accurately each drone would require at least three neighbors, and it may happen that during operation some drones do not meet this requirement. To overcome this issue, a weighted averaging between the drones of the network is implemented, allowing drones with insufficient neighbors to estimate the target's position accurately. By propagating the weighted average of the robot's estimated position from each drone across the network, it is possible to reach agreement on the overall estimate, giving more importance to drones with a greater number of neighbors and less importance to drones that do not have a sufficient number of neighbors available to provide an accurate estimate.

A. Paper organization

Sec. II describes how the field environment (Subsec. II-A), the robot (Subsec. II-B) and the drones (Subsec. II-C) have been modelled in terms of dynamic model, communication system and sensors. The TDOA-based localization approach is presented in Section Sec. III. In Sec. IV the most significant results are presented in detail with accompanying explanations. Finally Sec. V states the conclusions of the project.

II. System description and problem set-up

A. Environment model

The environment field $D(\boldsymbol{r})$ has been modelled as a simple Gaussian plume model since if a substance diffuses from a point source in an unbounded, wind-free environment, its

distribution profile is often considered as a bivariate Gaussian as shown in Equation 2. Venkatram and Thé [2003]

$$D(\boldsymbol{r}) = \frac{1}{\sqrt{(2\pi)^2 \cdot \det \boldsymbol{\Sigma}}} \cdot e^{\left(-\frac{1}{2}(\boldsymbol{r}-\boldsymbol{r}^0)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{r}-\boldsymbol{r}^0)\right)} \quad (1)$$

In the most straightforward case, where x and y coordinates are uncorrelated and have the same standard deviation, the preceding equation can be rewritten as follows:

$$D(\boldsymbol{r}) = q \cdot e^{-\frac{\|\boldsymbol{r}-\boldsymbol{r}^0\|^2}{2\cdot\sigma^2}} \quad (2)$$

The following assumption was then made:

***Assumption** 1:* (Field function)

1) The function $D(\cdot) \in C^2$ e.g. is twice continuously differentiable and has bounded curvature $|\nabla^2 D(\cdot)| \leq \mathbf{M}$. This assumption holds iff $\nabla D \in C^1$ *Lipschitz continuous* i.e. $\mathbf{M} = h\mathbf{I}$ implies $\|\nabla D(\boldsymbol{r}_1) - \nabla D(\boldsymbol{r}_2)\| \leq h\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|$. Note that $\mathbf{M}$ is not required to be positive definite (e.g. we are not assuming that $D(\cdot)$ is convex).
2) The function $D(\cdot) \in C^2$ is bounded by class-$\mathcal{K}_\infty$ functions $\kappa_1(\|\boldsymbol{r} - \boldsymbol{r}^0\|) \leq D(\boldsymbol{r}) \leq \kappa_2(\|\boldsymbol{r} - \boldsymbol{r}^0\|)$ and its gradient ∇D satisfies $\|\nabla D(\boldsymbol{r})\| \geq \kappa_3(\|\boldsymbol{r} - \boldsymbol{r}^0\|)$ for some class-κ_∞ function κ_3.
This assumption means that driving the field gradient to zero $\nabla D \to 0$ results in the output converging to the optimal $\boldsymbol{r}_t \to \boldsymbol{r}^0$ where the field reaches its maximum value $\gamma_{max} = \max_{r\in\mathbb{R}^2} D(\boldsymbol{r})$

B. Robot model

1) Robot dynamics: In the Euclidean plane $\mathbb{R}^2$, we consider a kinematic unicycle robot whose pose is completely described by $\boldsymbol{q} = [x\ y\ \theta]^T$ where (x, y) are the Cartesian coordinates and $\theta \in [0, 2\pi]$ is the forward orientation angle measured in radians counterclockwise from the horizontal axis of the absolute reference frame. Furthermore, the robot is equipped with a special Global Positioning System (GPS) sensor which provides the robot's position $[x\ y]$ via state estimation based on TDOA measurements by a drone network (described in Sec. III).

In general, the overall robot system can be described with the following *discrete-time*, *time-variant*, *non-linear* equations of the motion model and observation model:

$$\begin{aligned} \text{motion model:} \quad \hat{\boldsymbol{q}}_{k+1,k} &= g(\hat{\boldsymbol{q}}_{k,k}, \boldsymbol{u}_k) + \boldsymbol{\nu}_k \\ &= \mathbf{A}_k\hat{\boldsymbol{q}}_{k,k} + \mathbf{B}_k\boldsymbol{u}_k + \boldsymbol{\nu}_k \\ \text{observation model:} \quad \boldsymbol{z}_k &= h(\boldsymbol{q}_k) + \boldsymbol{\varepsilon}_k \\ &= \mathbf{C}_k\boldsymbol{q}_k + \boldsymbol{\varepsilon}_k \end{aligned} \quad (3)$$

where k is the discrete-time index and K its maximum, the function $g(\cdot)$ is the nonlinear motion model and the function $h(\cdot)$ is the nonlinear observation model. In the specific case of a unicycle motion model the variables in Equation 3 have the following meanings: $\boldsymbol{q} = [x\ y\ \theta]^T \in \mathbb{R}^{N=3}$ is the *system state*, $\mathbf{A} \in \mathbb{R}^{N\times N=3\times 3}$ is the *transition matrix*, $\boldsymbol{q}_0 \in \mathbb{R}^{N=3} \sim \mathcal{N}(\boldsymbol{q}_0, \mathbf{P}_0)$ is the *initial state*, $\mathbf{B} \in \mathbb{R}^{N\times U=3\times 2}$ is the *control matrix*, $\boldsymbol{u} = [v, w]^T \in \mathbb{R}^{U=2}$ is the *input* where $v \in \mathbb{R}$ and $\omega \in \mathbb{R}$ are respectively the linear and angular velocity of the unicycle robot, $\boldsymbol{\nu} = [\nu_x, \nu_y, \nu_\theta]^T \in \mathbb{R}^3 \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ is the additive Gaussian *process noise* that embeds both the uncertainty on the model and the odometry sensor noise with $\mathbf{Q} = \mathsf{E}\{\boldsymbol{\nu}\boldsymbol{\nu}^T\}$ the covariance matrix, $\boldsymbol{z} = [z_x, z_y] \in \mathbb{R}^{M=2}$ is the *measurement*, $\mathbf{C} \in \mathbb{R}^{M\times N=2\times 3}$ is the *observation matrix*, $\boldsymbol{\varepsilon} = [\varepsilon_x, \varepsilon_y] \in \mathbb{R}^{M=2} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ is the Gaussian *measurement noise* that embeds the uncertainty of the GPS measurement with $\mathbf{R} = \mathsf{E}\{\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T\}$ the covariance matrix.

Furthermore, we have that

$$\mathbf{A}_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{B}_k = \begin{bmatrix} \cos(\theta_k) & 0 \\ \sin(\theta_k) & 0 \\ 0 & 1 \end{bmatrix} \quad \mathbf{C}_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad (4)$$

By definition, the unicycle robot model is underactuated (i.e., three state variables and only two control inputs) and has the nonholonomic motion constraint of no sideway motion, i.e. $[-\sin\theta\ \ \cos\theta\ \ 0]\dot{\boldsymbol{q}} = 0$. We assume that the robot is only allowed to go forward with a constant speed ($v \geq 0$); angular velocity, on the other hand, can vary ($|\omega| \leq \bar{\omega}$).

Moreover, the following subscript notation will be employed:

$\boldsymbol{q}_k$	is the true pose of the robot
$\boldsymbol{z}_k$	is the pose measured value of the robot at time k
$\hat{\boldsymbol{q}}_{k,k}$	is the estimate of $\boldsymbol{q}$ at time k (the estimate is made after taking measurement $\boldsymbol{z}_k$)
$\hat{\boldsymbol{q}}_{k+1,k}$	is the estimate of the future state $(k+1)$ of $\boldsymbol{q}$ made at time k. In other words $\hat{\boldsymbol{q}}_{k+1,k}$ is a *predicted* state or extrapolated state
$\hat{\boldsymbol{q}}_{k-1,k-1}$	is the estimate of $\boldsymbol{q}$ at time $k-1$ (the estimate is made after taking the measurement $\boldsymbol{z}_{k-1}$)
$\hat{\boldsymbol{q}}_{k,k-1}$	is a *prior* prediction - the estimate of the state at time k. The prediction is made at time $k-1$
$\mathbf{P}_{k,k}$	is a covariance matrix that describes the squared uncertainty of an estimate
$\mathbf{P}_{k+1,k}$	is a covariance matrix that describes the squared uncertainty of a prediction

The following noises assumptions have been made:

***Assumption** 2:* (Noises) The noise random variables and the initial state knowledge are all assumed to be uncorrelated with one another (and with themselves at different time steps), i.e. $\mathsf{E}\{(\boldsymbol{\nu}_k - \boldsymbol{\mu}_\nu)(\boldsymbol{\varepsilon}_j - \boldsymbol{\mu}_\varepsilon)^T\} = 0\ \forall k, j$, furthermore we have assumed *wide sense whiteness* property on the noises.

2) Robot state estimation: Given the nonlinear nature of the motion model, the state estimation of the robot is obtained through the implementation of an EKF. EKF deals with nonlinearity in the systems by means of first-order approximation, i.e., linearization around the working point. Hence, we linearize the motion and observation models in Equation 3 about the current state estimate. The EKF is based on the following five underlying equations:

- Two prediction equations:
  - State extrapolation equation - estimating the future state based on the known present estimation

$$\hat{\boldsymbol{q}}_{k+1,k} = g(\hat{\boldsymbol{q}}_{k,k}, \boldsymbol{u}_k) \quad (5)$$

– Covariance extrapolation equation - the measure of uncertainty in the prediction

$$\mathbf{P}_{k+1,k} = \mathbf{J}_{\mathbf{g}_k}\mathbf{P}_{k,k}\mathbf{J}_{\mathbf{g}_k}^T + \mathbf{Q}_k \tag{6}$$

- Two update equations:
  – Kalman gain equation - required for the update equations. It is a "weighting" parameter for the measurement and the past estimation.

$$\mathbf{K}_k = \mathbf{P}_{k,k-1}\mathbf{J}_{\mathbf{h}k}^T(\mathbf{J}_{\mathbf{h}k}\mathbf{P}_{k,k-1}\mathbf{J}_{\mathbf{h}k}^T + \mathbf{R}_k)^{-1} \tag{7}$$

  – State update equation - estimating the current state based on the known past estimation and present measurement.

$$\hat{\boldsymbol{q}}_{k,k} = \hat{\boldsymbol{q}}_{k,k-1} + \mathbf{K}_k(\boldsymbol{z}_k - h(\hat{\boldsymbol{q}}_{k,k-1})) \tag{8}$$

  – Covariance update equation - the measure of uncertainty in our estimation in Joseph form

$$\mathbf{P}_{k,k} = (\mathbf{I}-\mathbf{K}_k\mathbf{J}_{\mathbf{h}k})\mathbf{P}_{k,k-1}(\mathbf{I}-\mathbf{K}_k\mathbf{J}_{\mathbf{h}k})^T+\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \tag{9}$$

where

$$\mathbf{J}_{\mathbf{g}_k} = \frac{\partial g(\boldsymbol{q},\boldsymbol{u},\boldsymbol{\nu})}{\partial \boldsymbol{q}}\Big|_{\hat{\boldsymbol{q}}_{k,k},\boldsymbol{u}_k,\mathbf{0}} = \begin{bmatrix} 1 & 0 & -v_k\cdot\sin\hat{\theta}_{k,k} \\ 0 & 1 & v_k\cdot\cos\hat{\theta}_{k,k} \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

$$\mathbf{J}_{\mathbf{h}k} = \frac{\partial h(\boldsymbol{q},\boldsymbol{\varepsilon})}{\partial \boldsymbol{q}}\Big|_{\hat{\boldsymbol{q}}_{k,k-1},\mathbf{0}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

A noteworthy aspect of unicycle robots is the absence of an exteroceptive sensor for angle estimation, such as a magnetometer. This results in a non-direct update of the orientation angle θ, in contrast to the x and y coordinates, which are updated indirectly through the use of GPS. However, during the prediction step of the EKF (Equation 6), the state covariance matrix $\mathbf{P}$ is computed using the Jacobian matrix $\mathbf{J}_\mathbf{g}$. The principal outcome of Equation 10 is a non-diagonal matrix $\mathbf{P}$, which implies that the x and y coordinates are correlated to θ. Consequently, in the update step (Equation 9), upon the occurrence of the GPS measurement, it is possible to enhance the estimation of the angular coordinate.

The block diagram shown in Figure 1 provides a detailed description of the developed Kalman Filter.

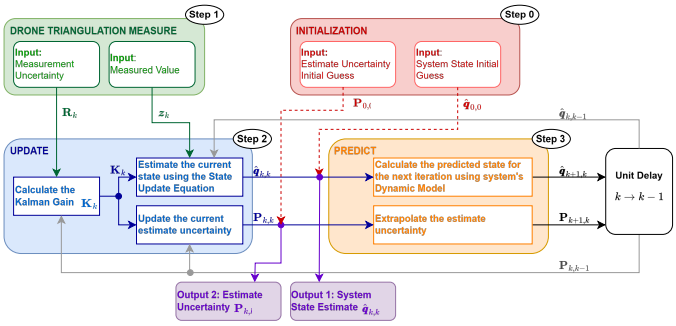

Figure 1: Robot's EKF state estimation block diagram

In Initialization (Step 0) two parameters are provided: $\hat{\boldsymbol{q}}_{0,0}$ and $\mathbf{P}_{0,0}$; in Measurement (Step 1) are provided $\boldsymbol{z}_k$ and $\mathbf{R}_k$; in Step Update (Step 2) the state update process is performed providing as output $\hat{\boldsymbol{q}}_{k,k}$ and $\mathbf{P}_{k,k}$; in Predict (Step 3) the prediction process is performed returning as output: $\hat{\boldsymbol{q}}_{k+1,k}$ and $\mathbf{P}_{k+1,k}$.

In its initial state, the robot is unable to ascertain its position in relation to the global reference system. Consequently, the position in the initial state is placed at the axis origin $\hat{\boldsymbol{q}}_{0,0} = [0,0,0]$. Then, in order to simulate the initial unknowledge the covariance is set to high values $\mathbf{P}_{0,0} = \mathbf{I}_{3\times3}\cdot 100$.

On occasion, the sensor measurements are lost as a consequence of noisy communication channels, interferences, equipment malfunctions or other anomalies. In such instances, the measurement variance in Equation 7 is set to exceedingly high values (e.g. $\mathbf{R}_k = \mathbf{I}_{2\times2}\cdot 10^{10}$) resulting in a Kalman gain that is approximately equal to zero ($\mathbf{K}_k \approx \mathbf{0}_{3\times2}$). This results that the current update estimation covariance, in Equation 9, being equal to the prior extrapolated state covariance (e.g. $\mathbf{P}_{k,k} = \mathbf{P}_{k,k-1}$), with the extrapolated state covariance in Equation 6 increasing due to process noise and linearity model approximation $\mathbf{J}_{\mathbf{g}_k}$ (e.g. $\mathbf{P}_{k+1,k} = \mathbf{J}_{\mathbf{g}_k}\mathbf{P}_{k,k}\mathbf{J}_{\mathbf{g}_k}^T + \mathbf{Q}_k = \mathbf{J}_{\mathbf{g}_k}\mathbf{P}_{k,k-1}\mathbf{J}_{\mathbf{g}_k}^T + \mathbf{Q}_k$). Consequently, the absence of the requisite measurement results in a transient divergence of the Kalman filter.

3) Robot motion control implementation: In order for the robot to be able to achieve its goal, it is necessary to inject a specific control input $\boldsymbol{u}$ into the dynamic equation (Equation 3). Although most of the literature addresses the gradient search problem using algorithms such as the Extremum Gradient Seeking (ESC) which is an adaptive control method in which input-output characteristics are optimized without requiring any explicit input-output relationship Danielson et al. [2023] Jabeen et al. [2023] Biyik and Arcak [2007] Zhang et al. [2021], it has been decided to face the implementation of the high-level controller replicating the method implemented by Matveev et al. [2011], who proposed a control algorithm that does not employ gradient estimation and is non-demanding with respect to both computation and motion. The objective is to change only the angular velocity according to the control algorithm shown in Equation 11 in order to drive the robot into the R^*-neighbourhood $V^* = \{\boldsymbol{r} : \|\boldsymbol{r} - \boldsymbol{r}^0\| \leq R^*\}$ and keep the robot within this neighbourhood:

$$\omega(t) = \bar{\omega}\cdot\operatorname{sgn}\left(\dot{d}(t) - v_*\right) \tag{11}$$

where $v_* > 0$ is a controller parameter, and $\dot{d}(t)$ is the rate of change of the field measurement. The parameter v_* plays a critical role as a tunable parameter that influences the robot's movement towards the extrema of the unknown field. This means that the robot adjusts its direction based on whether the rate of change $\dot{d}(t)$ is greater or less than the desired growth rate v_*. When $\dot{d}(t)$ exceeds v_*, the robot turns left ($\bar{\omega}$ is a positive value), in the opposite case, a right turn is executed. Therefore the choice of v_* affects the convergence of the robot to the maximum of the field and the overall performance of the navigation algorithm. If v_* is too small, the robot may take longer to converge to the desired vicinity of the maximum. On the other hand, if the v_* is too large it may lead to less accurate

convergence or failure to remain in the neighbourhood of the maximum point.

Without going through the mathematical assumption and theorem proofs of the paper, assuming an isotropic field function $D(\boldsymbol{r})$ that is twice continuously differentiable, and its maximum $\gamma_{\max}$ being attained at a unique point $\boldsymbol{r}_0$ as we assume in Ass.1, given an estimate q_- of the field strength of Equation 2 such that $q \geq q_- > 0$, the upper bound of the v^* value results from minimization of the right-hand side of Equation 12 over an estimated range of the dispersion $\sigma \in [\sigma_-, \sigma_+]$:

$$0 < v_* < \frac{q_- v}{\sigma^2} \begin{cases} +\infty & \text{if } \sigma < R_-; \quad \text{otherwise} \\ \frac{R_- \mathrm{e}^{-\frac{-R_-^2}{2\sigma^2}}}{\sqrt{1+\frac{RR_-}{R_- - R}\left[\frac{1}{R_-} - \frac{R_-}{\sigma^2}\right]}} \end{cases} \tag{12}$$

where $R_- = R^* - 2R$, this solution works under the hypothesis of $R^* > 3 \cdot R$ with $R = \frac{v}{\omega}$ the robot turning radius.

Therefore, under a partial knowledge of the field and the assumption previously made, Matveev et al. [2011] state that the value of $v*$ chosen inside the range Equation 12 guarantee that the robot can reach, in a finite time, the R^*-neighbourhood of the field peak. In spite of Matveev's guidance, the maximum value of v^* was always taken as a reference. From the simulations carried out, it was noted that too small values of v^* (i.e. desired field growth too low) causes the robot to remain stable on a certain field isoline and consequently the robot never reaches the target.

C. Drone model

1) Drone initialization: Let's denote with *ENV_SIZE* the size of the environment. A number N of drones is randomly sampled between a specified maximum and minimum number. Each drone is randomly initialized within the environment, defining a circular region of radius $r <$ *ENV_SIZE*, ensuring that a minimum distance between each drone is respected. This allows to

- distribute the drones more uniformly, ensuring good coverage of most of the environment within which the robot moves
- have drones with different distance between them reducing the ambiguity of TDOA measurements
- position the drones at different heights increasing the geometric diversity of the measurements

Each drone is in communication with a subset $M \subseteq N$ of drones, based on a communication radius, which is chosen to be equal to the radius r of the circular region within which the drones are positioned. Neighboring drones are then able to communicate and exchange the information necessary to estimate the robot's position. The drone network forms therefore a connected, but not complete, graph. This can be seen in Figure 2, where an example graph is presented with the connections between the drones.

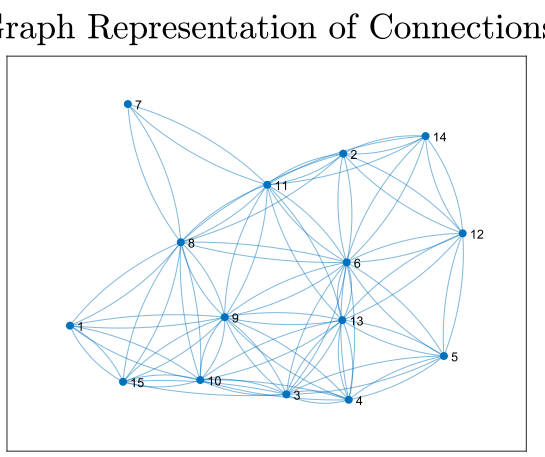

Figure 2: Example of drone network connectivity

The drones are initialized in such a way that each drone has at least one neighbor, however, when moving, this condition may cease to exist, and it may happen that at least one drone disconnects from the graph, no longer being able to communicate with the other drones and, consequently, no longer participating in the estimation of the robot's position, which will lose accuracy. If such a situation occurs, the system is designed in such a way that the drone that loses the connection continues to save the last estimate taken in its memory, so that, if the drone regains the connection, it can return to participate in the estimation phase.

2) Drone dynamics: Unlike the robot, each drone moves in the three-dimensional Cartesian plane and therefore has the z coordinate that is different from zero. The real position of each drone within the environment is provided by the following generalized state coordinates: $\boldsymbol{s} = [x\ y\ z]^T$. Each drone moves within the environment according to a linear dynamic model described by the following state-space representation:

$$\boldsymbol{s}_{k+1} = \mathbf{A}_k \boldsymbol{s}_k + \mathbf{B}_k \boldsymbol{u}_k + \mathbf{W}_k \boldsymbol{w}_k \tag{13}$$

where $\boldsymbol{u} = [v_x, v_y, v_z]^T \in \mathbb{R}^{N=3}$ is the *control input* expressed as linear velocity along x, y and z axis, $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T \in \mathbb{R}^3 \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ is the *process noise*, $\mathbf{A} \in \mathbb{R}^{N \times N=3 \times 3}$ is the *state transition matrix*, $\mathbf{B} \in \mathbb{R}^{N \times N=3 \times 3}$ is the *control matrix* and $\mathbf{W}$ is the *process noise matrix*, respectively in the form:

$$\mathbf{A}_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{B}_k = \begin{bmatrix} dt & 0 & 0 \\ 0 & dt & 0 \\ 0 & 0 & dt \end{bmatrix} \quad \mathbf{W}_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{14}$$

where dt is the time step of the simulation. The following assumption has been made on the process noise:

***Assumption** 3:* (Process noise) The process noise is assumed to be an uncorrelated, zero mean, white Gaussian noise with: $\mathsf{E}\{\boldsymbol{\omega}_k \boldsymbol{\omega}_l^T\} = Q\delta_{kl}\ \forall k, l$, where δ_{kl} is the Kronecker delta.

3) Drone motion control implementation: In typical tracking applications, several stations/anchors are placed on the ground and kept stationary at that specific location, in order to track the movements of a moving target within the environment. Drones could act as stationary stations positioned at a non-zero z altitude. To ensure greater realism of the application, the drones are mobile and are implemented with velocity controls with respect to the three Cartesian axes x, y and z. These controls enter in the dynamic law expressed in Equation 13 and are calculated based on information known to the drones themselves, in particular the estimate of the robot's position at each instant of time. Once two consecutive estimates of the robot's position are computed, the controls are given based on the following procedure:

Procedure COMPUTE DRONES CONTROLS
$\boldsymbol{v}_k = [\hat{x}_k - \hat{x}_{k-1}, \hat{y}_k - \hat{y}_{k-1}, 0]$
if ${}^{z}s_k^i < z_{min}$ **then**
  ${}^{x,y}\boldsymbol{u}_k^i = u_{max} \frac{{}^{x,y}\boldsymbol{v}_k}{||{}^{x,y}\boldsymbol{v}_k||}$
  ${}^{z}\boldsymbol{u}_k^i = {}^{z}u_k = \text{const.}$
else
  $\boldsymbol{u}_k^i = u_{max} \frac{\boldsymbol{v}_k}{||\boldsymbol{v}_k||}$
end if
End Procedure

where k is the time instant, i is the i-th drone in the system, $\hat{x}_{k/k-1}$/$\hat{y}_{k/k-1}$ are the xy-coordinates of the robot estimated by the drone network at time k or $k-1$ and ${}^{z}s_k^i$ is the z-coordinate of the i-th robot in the network. As can be seen the movement along z-axis is controlled only by the noise affecting the dynamics of the drones (ω_z). Since noise can drive the z-coordinate to negative values, especially for drones starting at low altitudes, a correction is applied whenever a drone is lower than a limit height, which is set at 1 m. A control velocity in x and y is provided to ensure that each drones follow a similar trajectory to the one followed by the robot, maintaining approximately the same initial distances. Recalling subsubsection II-C1, if a drone is disconnected from the graph, it is given zero velocity along the three axis, and therefore its movement is controlled only by the noise on the dynamic model.

III. DRONE LOCALIZATION IMPLEMENTATION

A. Measurement model

TDOA-based localization is performed using a network of N identical agents/drones. Each drone is able to exchange information only with its neighboring drones. In particular, each drone measures the TDOA of the signal emitted by the moving target between itself and all its neighbors. The measurement model of the i-th drone in the system can then be expressed as

$$\boldsymbol{z}_k^i = h^i(\boldsymbol{q}_k) + \boldsymbol{\delta}_k^i \tag{15}$$

where $\boldsymbol{q}_k = [x_k, y_k, z_k]$ is the state of the robot and $\boldsymbol{\delta}_k^i \in \mathbb{R}^{M_i} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ is the *measurement noise* of drone i, with M_i being the set of all neighboring drones of drone i. Since the robot moves in the 2D space, we already know that its z-coordinate is constrained to be zero. The following assumption has been made on the measurement noise:

Assumption 4: (Measurement noise) The measurement noise is assumed to be an uncorrelated, zero mean, white Gaussian noise with: $\mathsf{E}\{\boldsymbol{\delta}_k^i \boldsymbol{\delta}_l^{j^T}\} = R^i \delta_{kl} \delta_{ij}\ \forall k, l, i, j$.
Let's denote with $\boldsymbol{s}_k^i$ the position of the i-th drone at time k, and with $\boldsymbol{s}_k^{i,j}$ the position of the j-th neighbor of drone i at time k, then the measurement model can be expressed as

$$\boldsymbol{h}^i(\boldsymbol{q}_k) = \begin{bmatrix} h_1^i(\boldsymbol{q}_k) \\ h_2^i(\boldsymbol{q}_k) \\ \dots \\ h_{M_i}^i(\boldsymbol{q}_k) \end{bmatrix} = \begin{bmatrix} ||\boldsymbol{q}_k - \boldsymbol{s}_k^i|| - ||\boldsymbol{q}_k - \boldsymbol{s}_k^{i,1}|| \\ ||\boldsymbol{q}_k - \boldsymbol{s}_k^i|| - ||\boldsymbol{q}_k - \boldsymbol{s}_k^{i,2}|| \\ \dots \\ ||\boldsymbol{q}_k - \boldsymbol{s}_k^i|| - ||\boldsymbol{q}_k - \boldsymbol{s}_k^{i,M_i}|| \end{bmatrix} \tag{16}$$

where $||\boldsymbol{q}_k - \boldsymbol{s}_k^i||$ is the TOA measurement of the signal emitted by the robot and received by the i-th drone. Assuming the drones are all synchronised with each other, by taking the difference between two TOA measurements, a TDOA measurement can be obtained, and the measurement model is defined. The measurement function in Equation 16 can be linearized by means of the Jacobian function:

$$\mathbf{H}_k^i = \begin{bmatrix} \frac{\partial h_1^i}{\partial\, {}^{x}q_k} & \frac{\partial h_1^i}{\partial\, {}^{y}q_k} & \frac{\partial h_1^i}{\partial\, {}^{z}q_k} \\ \frac{\partial h_2^i}{\partial\, {}^{x}q_k} & \frac{\partial h_2^i}{\partial\, {}^{y}q_k} & \frac{\partial h_2^i}{\partial\, {}^{z}q_k} \\ \dots & \dots & \dots \\ \frac{\partial h_{M_i}^i}{\partial\, {}^{x}q_k} & \frac{\partial h_{M_i}^i}{\partial\, {}^{y}q_k} & \frac{\partial h_{M_i}^i}{\partial\, {}^{z}q_k} \end{bmatrix} \tag{17}$$

where the single elements can be rewritten as:

$$\begin{aligned} \frac{\partial h_j^i}{\partial\, {}^{x}q_k} &= \frac{{}^{x}q_k - {}^{x}s_k{}^{i}}{||\boldsymbol{q}_k - \boldsymbol{s}_k^i||} - \frac{{}^{x}q_k - {}^{x}s_k{}^{i,j}}{||\boldsymbol{q}_k - \boldsymbol{s}_k^{i,j}||} \\ \frac{\partial h_j^i}{\partial\, {}^{y}q_k} &= \frac{{}^{y}q_k - {}^{y}s_k{}^{i}}{||\boldsymbol{q}_k - \boldsymbol{s}_k^i||} - \frac{{}^{y}q_k - {}^{y}s_k{}^{i,j}}{||\boldsymbol{q}_k - \boldsymbol{s}_k^{i,j}||} \\ \frac{\partial h_j^i}{\partial\, {}^{z}q_k} &= \frac{{}^{z}q_k - {}^{z}s_k{}^{i}}{||\boldsymbol{q}_k - \boldsymbol{s}_k^i||} - \frac{{}^{z}q_k - {}^{z}s_k{}^{i,j}}{||\boldsymbol{q}_k - \boldsymbol{s}_k^{i,j}||} \end{aligned} \tag{18}$$

where ${}^{x}s_k{}^{i,j}$, ${}^{y}s_k{}^{i,j}$ and ${}^{z}s_k{}^{i,j}$ are the xyz-coordinates of the j-th neighbor of drone i at time k, whereas ${}^{x}q_k$, ${}^{y}q_k$ and ${}^{z}q_k$ are the xyz-coordinates of the robot at time k.

B. Extended Kalman Filter

Each drone implements its own EKF to estimate the state of the robot. The EKF is a two-step recursion consisting of time-update and measurement-update. Initially at k=0 they all start from an initial guess, which is chosen to be the centre of the environment, which corresponds to the origin of the Cartesian plane ($\hat{\boldsymbol{q}}_0 = [\hat{x}_0, \hat{y}_0, \hat{z}_0]^T = [0, 0, 0]^T$). At the beginning, given the uncertainty on the real position of the robot, the covariance matrix is initialized at high values, i.e. $\overline{\mathbf{P}}_0^i = \mathbf{I}_{3\times3} \cdot 100$. In the time-update step, since each drone runs the EKF to estimate the motion of the robot, the covariance matrix is computed as the sum between the covariance matrix of the previous estimation step and the covariance matrix related to the process noise of the robot. It's important to understand that the matrix $\mathbf{Q}$ used within the EKF of the drones is different from the covariance matrix

$\mathbf{Q}$ defined in the dynamic model of the drones themselves, because the former accounts for the uncertainty in the dynamic model of the estimation target, i.e. the robot, whereas the latter describes the uncertainty on the dynamic model of the drones, and affects only their motion. It's also important to note that the relationship between the TDOA measurements and the position of the robot involves the distances from the robot to each drone. Since distance is related to time via the speed of signal propagation, the relationship between the time differences and the position coordinates involves the square root of a sum of squares. This implies the measurement model is non-linear and a linearization around the current estimate is necessary:

- Time-update:

$$\begin{aligned} \hat{\boldsymbol{q}}^i_{k,k-1} &= \overline{\boldsymbol{q}}^i_{k-1} \\ \mathbf{P}^i_{k,k-1} &= \overline{\mathbf{P}}^i_{k-1} + \mathbf{Q}^i_k \end{aligned} \tag{19}$$

- Measurement-update:

$$\begin{aligned} \mathbf{K}^i_k &= \mathbf{P}_{k,k-1}\mathbf{H}^{i^T}_k (\mathbf{H}^i_k \mathbf{P}^i_{k,k-1}\mathbf{H}^{i^T}_k + \mathbf{R}^i_k)^{-1} \\ \hat{\boldsymbol{q}}^i_{k,k} &= \hat{\boldsymbol{q}}^i_{k,k-1} + \mathbf{K}^i_k(\boldsymbol{z}^i_k - h^i(\hat{\boldsymbol{q}}^i_{k,k-1})) \\ \mathbf{P}^i_{k,k} &= (\mathbf{I} - \mathbf{K}^i_k\mathbf{H}^i_k)\mathbf{P}^i_{k,k-1}(\mathbf{I} - \mathbf{K}^i_k\mathbf{H}^{i^T}_k)^T \\ &\quad + \mathbf{K}^i_k\mathbf{R}^i_k\mathbf{K}^{i^T}_k \end{aligned} \tag{20}$$

C. Weighted-averaging consensus

After each drone has executed its own EKF, a weighted average between the estimated state by the ith-drone and the estimated state by all its neighboring drones is performed. The same happens for the estimated covariance matrix:

$$\begin{aligned} \overline{\boldsymbol{q}}^i_k &= \sum_{j=1}^{N} w_{ij}\hat{\boldsymbol{q}}^j_{k,k} \\ \overline{\mathbf{P}}^i_k &= \sum_{j=1}^{N} w_{ij}\mathbf{P}^j_{k,k} \end{aligned} \tag{21}$$

The weights are based on the number of neighbors (M) of each drone. In this sense, drones with a larger number of neighbors are given more importance since they have collected a larger amount of measurements from other drones within the same environment:

$$w_{ij} = \begin{cases} 0, & \text{if } j \notin M_i \cup i \\ \frac{M_j}{\sum_{l \in M_i \cup i} M_l}, & \text{otherwise} \end{cases} \tag{22}$$

A simple consensus algorithm is executed for the drones to reach agreement on the estimated state of the robot and for the drone system to provide a single estimate, which can then be used as a measurement by the robot in its EKF to estimate the actual trajectory. In this step the drones continue to update their estimates iteratively according to the weighted average principle previously discussed in Equation 21. In this way the estimates of each drone are propagated among the drone network until convergence to the same value among all the drones is achieved. Convergence is guaranteed by iterating the process for a sufficiently high number of iterations such that the difference between the estimates is no longer significant. By applying this method a new estimate $\hat{\boldsymbol{q}}_k$ and $\mathbf{P}_k$ is generated, which is then used as new starting point in the EKF of each drone for the next EKF iteration.

IV. Results

In order to validate the work, the simulation of the robot and drone localization system was performed with the parameters reported in Table I .

Table I: Constant simulation parameters

`dt` $= 0.1$s	Simulation time step
`max_iter`=10^4	Maximum number of iterations
`ENV_SIZE`= 60m	Size of the environment
`ENV_STEP`= 200	Number of steps of the environment.
`q0`= $[-25m, 25m, 30°]$	Initial position of the robot
`std_dyn_xy`= 0.5m	Robot Standard deviation of the dynamic noise for x, y coordinates
`std_dyn_theta`= 1°	Robot Standard deviation of the dynamic noise for the θ coordinate
`N_min_agents`= 8	Minimum number of drone agents
`N_max_agents`= 16	Maximum number of drone agents
`radius`= 40m	Radius of region for agent's initialization
`communication radius`= 40m	Radius for drones communication
`std_drones`= 5m	Standard deviation of the GPS noise, i.e. the drone measurement model noise

The simulation has been performed with the robot constant speed $v = 0.7$m/s and time-varying angular velocity limited by $\bar{\omega} = 0.8$rad/s, the minimal turning radius is therefore $R = 0.875$m. We assumed to drive the robot within a distance of $R_* = 3$m (the ass. $R* > 3R = 2.6250m$ is respected) from the field maximum location. The field examinated as the Gaussian distribution of $D(x, y) = 10 \cdot e^{(x-8)^2+(y-5)^2/(2\sigma^2)}$ with $\sigma^2 = 300$. The chosen exact field is unknown to the robot but knowing the topology of the field some assumption can be made e.g. the field strength lower bound is assumed equal to $q_- = 5$ ($q \leq q_- \leq 0$), the estimated range of dispersion is assumed equal to $\sigma^2_- = 100$, $\sigma^2_- = 500$. From Equation 12 the value of v^* can be chosen within the range $0 < v^* < 0.0481$, consequently $v^* = 0.048$ was chosen. the robot was started from an arbitrary initial position $q_0 = [-20m, 20m, 30°]$.

During the simulation, a malfunction with the 'GPS drone system' was also simulated from iteration 100 to iteration 150 (i.e. for 5 seconds from time 10s to time 15s).

The real and the estimated position of the robot is shown in Figure 3 while Figure 4 shows the robot coordinates $\boldsymbol{q}_k = [x, y, \theta]$ with their respective errors.

It was thus demonstrated that the robot employing the developed motion control algorithm is able to reach the centre of the field in 462 iterations (i.e. 46.2 s), utilising solely the value of the field at the current position and at the previous instant, as shown in Figure 3. Furthermore, the GPS drone system enables the robot to accurately determine its current position as also shown in Figure 4 where the robot coordinates $\boldsymbol{q}_k = [x, y, \theta]$ and their error are indicated. From the outset, the error in the x,y coordinates declines significantly, whereas the error in the steering angle, caused by an absence of a direct update from a exteroceptive sensor that estimates the angle, exhibits a less pronounced reduction. Conversely, it can

be clearly seen that from a certain instant of time onwards the absence of GPS connectivity results in a divergence in the robot's estimation. This malfunctioning can clearly be seen in Figure 5 where the absence of the GPS signal leads to a persistent increase in the standard deviation of the x,y coordinates due to the noise of the motion model and linear approximation of the model.

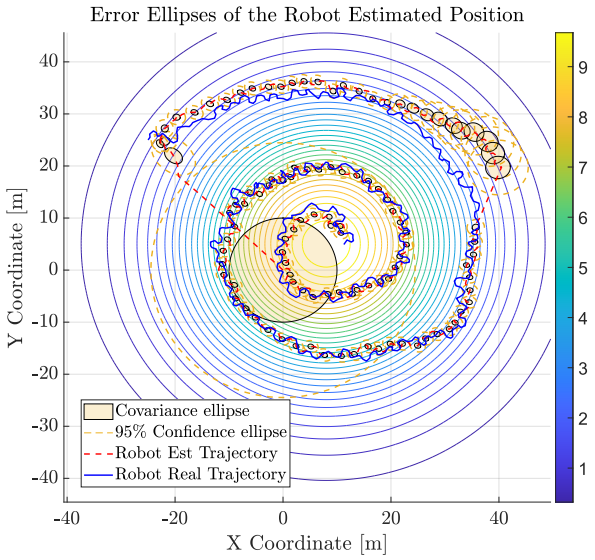

Figure 3: Robot real path and estimated path

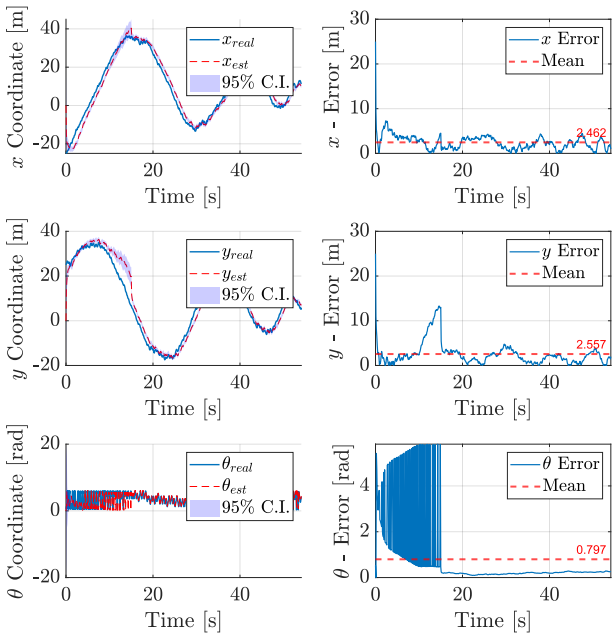

Figure 4: Robot real and estimated coordinates with respective errors. From top to bottom respectively the x,y and θ coordinates

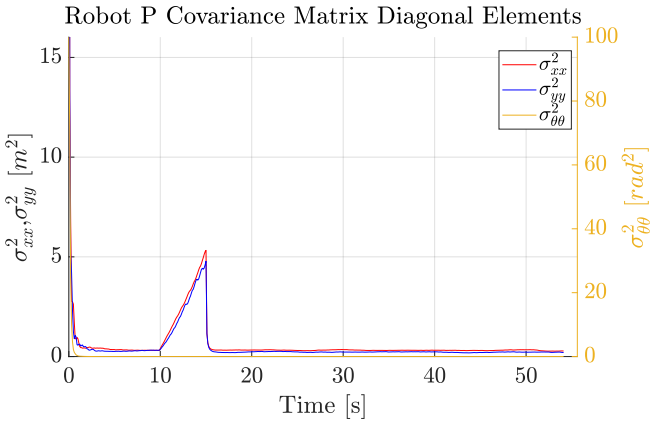

Figure 5: Robot $\mathbf{P}$ covariance matrix diagonal elements

A. *Further results*

1) Robot motion controller based on PID: Inspired by the control algorithm developed by Matveev et al. [2011], a second control algorithm based on a Proportional–Integral–Derivative (PID) controller for the variation of the parameter v^* was developed. As previously mentioned, the parameter v^* is crucial in achieving the goal, as it identifies the variation in field magnitude, but this value is hypothetically unknown, various assumptions have been made regarding the field, and there is no adaptability to maintain the desired performance. Consequently, via a PID controller, it can be varied according to the measured variation $\dot{d}$. This PID controller can still lead to divergences if the weights are not correctly chosen. It is therefore not necessary to make assumptions about the field, but about the controller weights. The following PID weight has been chosen $k_p = 0.1$ in order to respond gently to changes in $\dot{d}(t)$; $k_i = 0.05$ in order to help to eliminate steady-state error but slowly, which may affect how quickly the system can respond to persistent errors in field distribution; $k_d = 0.01$ in order minimize the effect of rate of $\dot{d}(t)$ change, thus reducing the potential for overshoot but also limiting the controller's ability to predict and preemptively adjust to dynamic changes in the environment. Therefore the new controller employ $v^*(k) = k_p \cdot \dot{d}(k) + I(k) + D(k)$, where $I(k) = I(k-1) + k_i \cdot \dot{d}(k) \cdot dt$ and $D(k) = K_d \cdot \frac{\dot{d}(k) - \dot{d}(k-1)}{dt}$. This lead to the result reported in Figure 6 and Figure 7

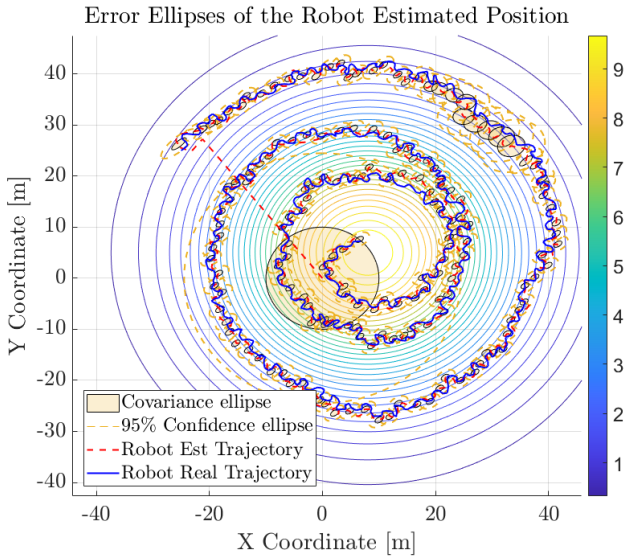

Figure 6: Robot real path and estimated path with PID

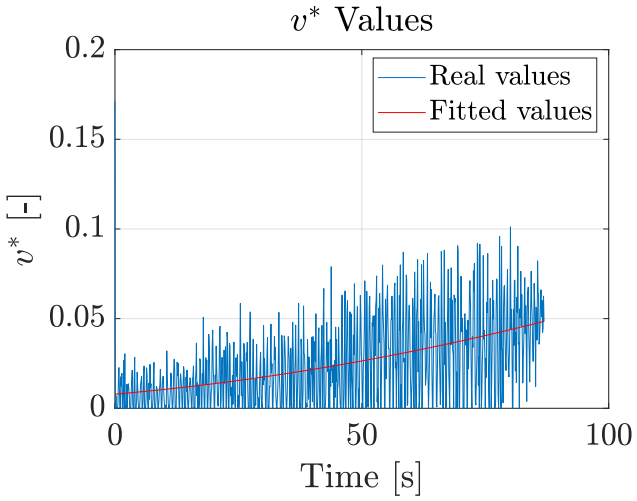

Figure 7: Robot v^* values that are constantly updated

V. Conclusions

A distributed TDOA-based localization scheme is presented. Multiple moving drones are used to collect TDOA measurements and generate estimates on the position of a robot moving within an environment, with the aim of moving from an initial random position to a final target position, which corresponds to the center of an unknown field, by generating appropriate controls to make the robot follow the gradient of the field. The estimated positions are used by the robot to update its predicted state within the EKF to generate an estimated trajectory of its own real trajectory, which is not known. The capability of the drone network to provide accurate measures of the real position of the robot is proved by ensuring a weighted average consensus across the network. This allows to have an accurate estimated trajectory allowing the robot to understand when the target position has been reached.

References

E. Biyik and M. Arcak. Gradient climbing in formation via extremum seeking and passivity-based coordination rules. In *2007 46th IEEE Conference on Decision and Control*, pages 3133–3138, 2007. doi: 10.1109/CDC.2007.4434735.

C. Danielson, S. A. Bortoff, and A. Chakrabarty. Extremum seeking control with an adaptive gain based on gradient estimation error. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 53(1):152–164, 2023. doi: 10.1109/TSMC.2022.3171132.

Y. Elor and A. M. Bruckstein. "robot cloud" gradient climbing with point measurements. *Theoretical Computer Science*, 547:90–103, 2014. ISSN 0304-3975. doi: https://doi.org/10.1016/j.tcs.2014.06.025. URL https://www.sciencedirect.com/science/article/pii/S0304397514004617.

O. Ennasr, G. Xing, and X. Tan. Distributed time-difference-of-arrival (tdoa)-based localization of a moving target. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 2652–2658. IEEE, 2016.

M. Jabeen, Q.-H. Meng, T. Jing, and H.-R. Hou. Robot odor source localization in indoor environments based on gradient adaptive extremum seeking search. *Building and Environment*, 229:109983, 2023. ISSN 0360-1323. doi: https://doi.org/10.1016/j.buildenv.2023.109983. URL https://www.sciencedirect.com/science/article/pii/S0360132323000100.

A. S. Matveev, H. Teimoori, and A. V. Savkin. Navigation of a unicycle-like mobile robot for environmental extremum seeking. *Automatica*, 47(1):85–91, 2011. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2010.10.003. URL https://www.sciencedirect.com/science/article/pii/S0005109810004176.

R. Suttner. Extremum seeking control for an acceleration controlled unicycle. *IFAC-PapersOnLine*, 52(16):676–681, 2019. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2019.12.040. URL https://www.sciencedirect.com/science/article/pii/S2405896319318701. 11th IFAC Symposium on Nonlinear Control Systems NOLCOS 2019.

A. Venkatram and J. Thé. *Introduction to Gaussian Plume Models*. 2003. URL https://apsi.tech/material/modeling/IntroductiontoGaussianPlumeModels.pdf.

T. Zhang, V. Qin, Y. Tang, and N. Li. Source seeking by dynamic source location estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2598–2605, 2021. doi: 10.1109/IROS51168.2021.9636841.