Automated source localization using a group of UAVs

Fabio Scaparro
Edoardo Colonna
Francesca Andreotti

September 2023

Abstract

UAVs are aerial veichles which flight is either controlled autonomously by on-board systems or piloted remotely. This type of robots results to be highly efficient during search and rescue missions, providing a support where the environment is extremely harsh, such as in avalanche victims searches scenarios.

In this project we proposed the implementation of a system which aims to reconstruct the victim position by means of a group of UAVs equipped with ARTVA receivers, exploiting the electromagnetic signals emitted by the ARTVA transmitter carried by the avalanche victim. The primary goal is to have a rendezvous of the UAV agents at the source location by following a consensus protocol, based on the exchange of information between the drones. For dealing with this purpose, the agents follow trajectories based on a PD controller, while the data provided by the UAVs network are collected in a decentralized model for being then used for the estimation of the transmitter location.

Contents

1 Introduction

The request of robotic support systems for human-based rescue operations is increasing more and more, due mainly to the several disaster areas in which these **Search and Rescue (S&R)** missions are performed. Related to the avalanches application context, rescue missions are characterized by specific peculiarities. One issue is constituted by the harsh environment where the disaster takes place, in which the scenario is constituted by unstable and irregular snow blocks, typically on steep slopes, making the human intervention difficult. Moreover, survival chances of the victim decrease rapidly with the increase of the rescue time, due to the possible hypothermia for the person buried under the snow. Therefore, S&R missions carried out just by humans could be quite slow and risky also for the rescuers themselves.

The tight requirements of the S&R missions, lead to imagine the development of an aerial robotic platform such as **unmanned aerial veichles (UAVs)**, carrying some device for accomplishing a faster localization of the victim position, while easily surveying the high mountain enviromnent.

One of the most common equipments used in avalanche setting is constituted by **ARTVA**, which is a sensor technology constitued by two easily switchable operating modes: a transmitter mode and a receiver mode. The ARTVA on the UAVs is in receiver mode, sensing and processing all the information about the electromagnetic field emitted by the ARTVA carried by the avalanche victim, which instead is in the transmitter mode.

The scientific community has promoted different activities regarding the application of S&R in avalanche settings, specifically in the context of the European project SHERPA [1], in which towards the years has been proposed and developed specific robotic technologies aiming to support professional alpine rescue teams in avalanche scenarios.

Since the previous works related to avalanche victim search by use of UAVs technology are based on centralized models for the collection of data, we focus on the implementation of a decentralized system, which aims to follow then a consensus protocol in order to bring all the agents (constituted by the UAV veichles) to agree on a common estimate for the position of the victim.

For these purposes, we propose the following project which aims to implement a system similar to the solution illustrated in [2] but in a decentralized control mode and by adding a consensus protocol with the aim of converging the drones at the same location, which should be the target position. The agents completes this task by sharing information between them in a controlled way and by following trajectories based on a PD controller on the current estimate. Moreover, we implement simulations of the overall process in a 3D environment, compared to the 2D- only environment implemented previously. Finally, for verifying the results obtained and the robustness of the system, we provide tests with respect to different connection graphs and a comparison between the centralized system and the decentralized approach.

First of all, in 2 we give a background explanation about the main notions of ARTVA technology and about decentralization modeling. In the 3, we show all the practical details about the project, while in 4 we illustrate experimental steps consisting in simulations performed in the MATLAB environment. In 5 we describe the results of our implementation choices and finally in 6 we give an overall view of the project, giving also our consideration about the whole work illustrated.

2 Background

2.1 Decentralized systems

Graphs are extremely powerful tools for encoding the information flowing among multiple agents, which are assumed to have individually limited capabilities of perception of the environment, communication and elaboration of data. More in details, every agent is limited in sensing/communication (information gathering), in computing power (information processing) and in available memory (information storage). Moreover, each robot (i.e., agent) has to analyse and process the information to run its local controller, impacting in terms of local computing power and memory. In order to handle these conditions, it is necessary to implement *decentralized controllers*, aiming to obtain that the size of the flowing information remains *constant* with respect to the number of agents, on each edge of the graph. In other words, the amount of data should grows linearly with the number of neighbors. It is possible to link graph theory to the study of multi-robot systems by seeing these ones as a collection of dynamical systems.

2.1.1 Algebraic Graph Theory

Algebraic Graph Theory field is based on the association to a graph of a specific set of matrices from which it is possible to derive several graph properties, obtaining in this way a complete and meaningful description of the network itself. For our purposes, we focus on **undirected graphs**, defined by a *vertex set* $\mathcal{V} = \{v_1, ..., v_N\}$ and an *edge set* $\mathcal{E} \subseteq [\mathcal{V}]^2$ [1], in which holds the following property: $(vi, vj) \in \mathcal{E} \Rightarrow (vj, vi) \in \mathcal{E}$. Moreover, given a node v_i we can define the *set* N_i *of all neighborgs of* v_i as $N_i = \{v_j \in \mathcal{V} | (vj, vi) \in \mathcal{E}\}$, from which it is possible to derive the degree [2] d_i of the *i-th* node itself.

Let's to consider the definition of the following matrices:

- *Adjacency matrix* $A \in \mathbb{R}^{N \times N}$, defined as follows:

$$\begin{cases} A_{ij} = 0 & if \quad (v_i, v_j) \notin \mathcal{E} \\ A_{ij} = 1 & if \quad (v_i, v_j) \in \mathcal{E} \end{cases}$$

  For undirected graphs, A is *squared* and *symmetric*. Moreover, since $A_{ii} = 0$ and $A_{ij} = A_{ji}$, it follows that $A = A^T$

- *Degree matrix* $\Delta \in \mathbb{R}^{N \times N}$, defined as[3]

$$\Delta = diag(d_i)$$

- *Incidence matrix* $E \in \mathbb{R}^{N \times |\mathcal{E}|}$, which is used for encoding the incidence relationship among edges and vertexes of the graph. By assigning an arbitrary orientation and an arbitrary labeling to the edges e, we can define E as follows

$$\begin{cases} E_{ij} = -1 & if \quad vertex \quad v_i \quad is \quad the \quad tail \quad of \quad edge \quad e_j \\ E_{ij} = 1 & if \quad vertex \quad v_i \quad is \quad the \quad head \quad of \quad edge \quad e_j \\ E_{ij} = 0 & otherwise \end{cases}$$

[1] $[\mathcal{V}]^2 = (v_i, v_j)$, with $i = 1, ...N, j = 1...N, i \neq j$
[2] $d_i = |N_i|$
[3] Or also as $\Delta = diag(\sum_{j=1}^{n} A_{ij})$

- *Laplacian matrix* $L \in \mathbb{R}^{N \times N}$, defined as[4]

$$L = \Delta - A$$

L matrix is *symmetric* and *positive semi-definite*. Thus, all its N eigenvalues λ_i are *real* and *non-negative*. Moreover, they are ordered as $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_N$. Since the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is *connected* (i.e., there exists a path joining any two vertexes in $\mathcal{V}$) if and only if $\lambda_2(\mathcal{G}) > 0$, the eigenvalue λ_2 is referred to as *connectivity eigenvalue*. Moreover, being $L\mathbb{1} = 0$, $\mathbb{1}$ is the eigenvector associated to λ_1 and $rank(L) = N - 1$

2.1.2 Consensus protocol

In graph theory context, the term *"consensus"* refers to the achievement of a general agreement by the graph vertices on the same state, which can be obtained by following specific protocols. Possible applications of consensus protocols are related to rendezvous operations, direction alignment, distributed estimation and synchronization of the agents (vertices) in the graph. The consensus protocol illustrated in this section holds assuming undirected graphs.
Let's to consider:

- N agents with an *internal state* $x_i \in \mathbb{R}$
- an *internal dynamics* for the state evolution (in this case, a single integrator $\dot{x_i} = u_i$)
- an interaction graph $\mathcal{G}$ having the agents as vertices

The problem consists of designing the *control inputs* u_i so that all the states x_i agree on the *same common value* $\bar{x}$, which is unspecified

$$\lim_{t \to \infty} x_i(t) = \bar{x}, \forall i \tag{1}$$

Moreover, the control inputs are designed so that each u_i exploits only the relative information based on the state of the neighborgs (i.e., relative sensing and decentralization)
The way in which the information flows across the agent is modeled by the graph $\mathcal{G}$. A solution for the illustrated problem consists of seeing the state u_i as the sum of all the differences of neighbors' states with respect to the state of the *i-th* agent.

$$u_i = \sum_{j \in \mathcal{N}_i} (x_j - x_i) \tag{2}$$

In compact form, for all the agents $u = -Lx$, and closing the loop, since $\dot{x} = u_i$ then $\dot{x} = -Lx$.
This closed-loop system has to solve the initial consensus requirement (1); moreover, since the Laplacian L represents the state-transition matrix in the closed-loop dynamics, the convergence to an arbitrary (but common) $\bar{x}$ is related to the properties of L itself. It can be proof [5] that the consensus protocol **converges** if the graph $\mathcal{G}$ is **connected**. This condition leads to obtain a fundamental result: all the agents x_i converge towards a **common value**, which is the **average of the initial state** x_0.

$$x_i \to \frac{\mathbb{1}^T x_0}{N}, \qquad \forall i$$

[4] Or also as $L = EE^T$

[5] Either by using the explicit solution of $\dot{x} = -Lx$ or by exploits Lyapunov Arguments

Formally, exploiting the definition of the *agreement subset* $\mathcal{A} \subseteq \mathcal{R}^N = span(\mathbb{1}) = \{x | x_i = x_j\}$, it is possible to assert that the consensus protocol makes the state $x(t) \to \mathcal{A}$. Furthermore, since $\mathbb{1}^T \dot{x} = -\mathbb{1}^T L x = 0$, the scalar quantity $\mathbb{1}^T x$ represents a constant motion of the closed-loop system

$$\mathbb{1}^T x(t) \equiv \mathbb{1}^T x_0 = const$$

Thus, the *centroid of the states* never changes over time. Finally, the *density of the graph* dictates the **rate of convergence** for the consensus protocol: the *more is connected* the graph, the *faster* the consensus convergence is[6]. [7]

2.2 ARTVA system: characterization and modeling

As mentioned in 1, The ARTVA equipment commercially available have two operating modes, in particular, it can work either as **receiver** or as **transmitter**, by means of a manual switch used to commute between the two. As first step before starting their activities, experienced skiers are equipped with the sensor, which is set to be used as *transmitter*, emitting an electromagnetic field. If an accident happens, remaining members of the team not buried by the avalanche, or rescuers in the disaster area, switch their devices to the *receiver mode*, which begin to capture information about the electromagnetic signal generated by the transmitter. The rescuers are trained to interpret these data to move towards the victim.
In this section, we go through the main physical principles of the ARTVA system aiming to derive a model of the signal vector field, in order to develop an automatic search algorithm as illustrated in 3

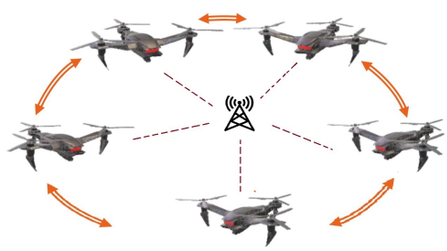

Figure 1: A network of UAVs exchanging information among them, while carrying an ARTVA receiver which is sensing the signal emitted by the ARTVA transmitter

2.2.1 Notation

Three Cartesian coordinate frames are defined as follows:

- $\mathcal{F}_i = (O_i, x_i, y_i, z_i)$ indicates the inertial frame with origin O_i, with the unitary vector x_i oriented towards geographic north, z_i, oriented opposite to the local gravity vector, and y_i oriented to create a right-hand frame
- $\mathcal{F}_t = (O_t, x_t, y_t, z_t)$ denotes the body right-hand frame associated to the transmitter (equipped by UAV)
- $\mathcal{F}_r = (O_r, x_r, y_r, z_r)$ denotes the body right-hand frame associated to the receiver (equipped by the victim)

We assume that the body frame of the drones coincides with the one of the receiver. The vector $p \in \mathbb{R}^3$ indicates the position of O_r relative to O_t, with

[6] It is related to the value of λ_2 (the smallest eigenvalue of L), which represents the degree of the connectivity of the graph: larger λ_2 is, faster is the convergence of the consensus
[7] P.R. Giordano Slides

$p = p_r - p_t$, while the position of O_r and O_t relative to O_i are indicated respectively by the vectors $p_r \in \mathbb{R}^3$ and $p_t \in \mathbb{R}^3$. Moreover, the rotation matrices from $\mathcal{F}_i$ to $\mathcal{F}_t$ and from $\mathcal{F}_r$ to $\mathcal{F}_i$ are indicated as R_t and R_r, while the rotation matrix R_{rt} denotes the relative rotation from $\mathcal{F}_t$ to $\mathcal{F}_r$. In conclusion, $h_t \in \mathbb{R}^3$ indicates the electromagnetic vector field described in $\mathcal{F}_t$.

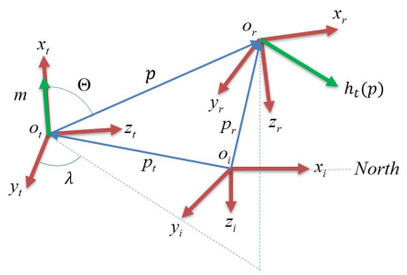

Figure 2: Definition of the reference frames

2.2.2 ARTVA mathematical model: transmitter mode

In transmission mode the ARTVA generates a magnetic field that is modeled as a dipole of amplitude $m \in \mathbb{R}_{>0}$ aligned with the x_t axes of $\mathcal{F}_t$.
The *mathematical model of the electromagnetic vector field* h_t at ${}^t p$ can be expressed as [3]

$$h_t({}^t p) = \frac{m}{4\pi ||p||^5} A_c({}^t p) \tag{3}$$

in which $A_c({}^t p)$ is defined as follow

$$A_c({}^t p) = \begin{bmatrix} 2p_x^2 - p_y^2 - p_z^2 \\ 3p_x p_y \\ 3p_x p_z \end{bmatrix}$$

and ${}^t p = col(p_x, p_y, p_z)$ indicates the vector p expressed in the frame $\mathcal{F}_t$ By expressing the model in polar coordinates $(\Theta, \lambda, ||p||)$ [8], it is possible to express h_t as

$$h_t(||p||, \Theta, \lambda) = \frac{m}{4\pi ||p||^3} A_p(\Theta, \lambda) \tag{4}$$

where

$$A_c({}^t p) = \begin{bmatrix} 2 - 3sin^2\Theta \\ 3cos\Theta sin\Theta cos\lambda \\ 3cos\Theta sin\Theta sin\lambda \end{bmatrix}$$

The flux lines expressed in the last illustrated equation results to be *symmetric* with respect to the transmitter x-axis.
Moreover, the *intensity of the electromagnetic field* $||h_t||$ is obtained from the previous definition of h_t in polar coordinates and can be exploited for computing the iso-power lines.

$$||h_t|| = \frac{m}{4\pi ||p||^3} \sqrt{1 + 3cos^2\Theta} \tag{5}$$

[8] where $\Theta := cos^{-1}(\frac{p_x}{\sqrt{p_x^2+p_y^2+p_z^2}})$, $\lambda := tan^{-1}(\frac{p_z}{p_y})$, $||p|| := \sqrt{p_x^2 + p_y^2 + p_z^2}$

2.2.3 ARTVA mathematical model: receiver mode

The ARTVA signal is received by means of three antennas directed along the receiver frame axes x_r, y_r and z_r. Assuming that the ARTVA receiver position p_r and orientation R_r in the inertial frame are known, the mathematical model of the magnetic field h_m sensed at the receiver location, is obtained from the projection of the vector h_t into the $\mathcal{F}_r$ frame.
In presence of noise ${}^r w(t)$ [9], the model is described as follows

$$h_m(p, R_{rt}, t) = R_{rt} h_t({}^t p) + {}^r w(t) \tag{6}$$

It is possible to denote h_n as the nominal electromagnetic field at the receiver, which is equal to h_m in absence of noise.

$$h_n(p, R_{rt}) = R_{rt} h_t({}^t p)$$

ARTVA model approximation Since the term $(1 + 3cos^2\Theta)^{\frac{1}{2}}$ introduces a significant complexity to the intensity of the EM field (5), a substantial simplification is given if this term is approximated with an equivalent one that belongs to the family of functions that are isomorphic to $(1+3cos^2\Theta)^{\frac{1}{2}}$, making (5) inversely proportional to a polynomial of p. By adopting the following approximation [4]

$$\frac{1}{\sqrt[3]{1 + 3cos^2\Theta}} \approx \frac{1}{a^2} cos^2\Theta + \frac{1}{b^2} sin^2\Theta \tag{7}$$

The known coefficients a and b are chosen in order to minimize the quadratic error with respect to $(1+3cos^2\Theta)^{-1/3}$. By applying this approximation, the norm of h_n becomes

$$||h_n|| \approx \frac{m}{4\pi} \left(\frac{a^2 b^2}{b^2 p_x^2 + a^2 (p_y^2 + p_z^2)} \right)^{\frac{3}{2}} \tag{8}$$

ARTVA transmitter localization by single UAV By considering the equation in 8, we can approximate the intensity of the signal emitted by the transmitter

$$y(p_r) = \Phi^T(p_r) x(p_t) \tag{9}$$

where

$$y = \left(\frac{m}{||h_n|| 4\pi} \right)^{\frac{2}{3}} (ab)^2 \tag{10}$$

$$\Phi(p_r) = col(p_{x_r}^2, 2p_{x_r}p_{y_r}, 2p_{x_r}p_{z_r}, p_{y_r}^2, 2p_{y_r}p_{z_r}, p_{z_r}^2, -2p_{x_r}, -2p_{y_r}, -2p_{z_r}, 1) \tag{11}$$

and in which:

- $p_r = (p_{x_r}, p_{y_r}, p_{z_r})$ is the vector of *known signal*
- $x(p_t) = col(\bar{m}_{11}, \bar{m}_{12}, \bar{m}_{13}, \bar{m}_{22}, \bar{m}_{23}, \bar{m}_{33}, \bar{p}_t, \rho) \in \mathbb{R}^{10}$ is the vector of the *unknown constants*

Moreover, $\bar{m}_{ij}$ are the entries of $\bar{M} = R_t diag(b^2, a^2, a^2) R_t^T$ with $\bar{M} = \bar{M}^T > 0, \bar{M} \in \mathbb{R}^{3x3}$, $\bar{p}_t := \bar{M} p_t$ and $\rho := p_t^T \bar{M} p_t$

The estimation for p_t can be provided by estimating the constant vector $x(p_t)$. Actually, the first six components of $x(p_t)$ constitute an estimate for $\bar{M}$, while $\bar{p}_t$

[9] ${}^r w(t) : \mathbb{R} \to \mathbb{R}^3$ indicates the ElectroMagnetic Interferences (EMI) expressed in the search plane frame

estimates $\bar{M}p_t$; therefore, p_t can be obtained by inversion of $\bar{M}$

ARTVA transmitter localization by multiple UAVs Now instead of having a single ARTVA receiver, we assume $n \in \mathbb{N}$ of them, singularly attached to an UAV [2] and a single ARTVA transmitter. The intensity of the signal emitted by the ARTVA transmitter can be approximated as follow

$$y(p_r, \tau_i) = \Phi^T(p_r)x(pt) + \delta(p_k - p_t) + v_t(p_k - p_t, \tau_i) \tag{12}$$

where

- $p_k(\tau_i) \in \mathbb{R}^3$ represents the inertial position of the *k-it* UAV [10]at the sample time τ_i with $k, i \in \mathbb{N}$ and $k = 1, \ldots, n$
- $p_t \in \mathbb{R}^3$ denotes the position of the transmitter
- $\delta(.) : \mathbb{R}^3 \to \mathbb{R}$ represents the *model mismatch* introduced by the approximation
- $v(.,.) : \mathbb{R}^3 \mathrm{x} \mathbb{R} \to \mathbb{R}$ is the contribution of y measurement noise

By the analysis of the contributions of δ and v [2], it is possible to show that the output is more informative when the ARTVA receiver becomes closer to the transmitter

2.2.4 Estimation of transmitter location using RLS

By assuming the presence of a communication network with static topology made of n-agents [2], the estimate of p_t can be obtained by considering as known the position of the *k-th* vehicle $p_k(\tau_i)$ and by exploiting all the information provided by the network and collected as data in the following centralized model

$$Y(\tau_i) = H^T(\tau_i)x(p_t) + \Delta(\tau_i) + V(\tau_i) \tag{13}$$

with

$$\begin{aligned} Y(\tau_i) &= col(y(p_1(\tau_i), \tau_i), ..., y(p_n(\tau_i), \tau_i)) \\ H^T(\tau_i) &= col(\Phi^T(p_1(\tau_i)), ..., \Phi^T(p_n(\tau_i))) \\ \Delta(\tau_i) &= col(\delta(p_1(\tau_i) - p_t), ..., \delta(p_n(\tau_i) - p_t)) \\ V(\tau_i) &= col(v(p_1(\tau_i) - p_t, \tau_i), ..., v(p_n(\tau_i) - p_t, \tau_i)) \end{aligned}$$

For the following considerations, we will consider the model 13 neglecting both the model mismatch introduced by the approximation $\Delta(\tau_i)$ and the contribution to y of the measurement noise $V(\tau_i)$ of each k-th agent

Recursive Least Squares (RLS) Let $\beta \in (0, 1)$ and $S(\tau_0) = S_0 = S_0^T > 0$ with $S_0 \in \mathbb{R}^{3x3}$. Assuming the availability of an accurate positioning system (ideal GPS), the constant vector x can be estimated by exploiting the model 13 and by applying the following *Recursive Least Squares (RLS)* [5]

$$\hat{x}(\tau_{i+1}) = \hat{x}(\tau_i) + S^{-1}(\tau_i)H(\tau_i)(Y(\tau_i) - H^T(\tau_i)\hat{x}(\tau_i)) \tag{14}$$

$$S(\tau_{i+1}) = \beta S(\tau_i) + H(\tau_i)H^T(\tau_i) \tag{15}$$

The position of the ARTVA transmitter is then estimated as follows

[10] $p_k = col(x_k, y_k, z_k)$

$$\hat{p}(\tau_i) = x^{-1}(\hat{x}(\tau_i)) \tag{16}$$

2.2.5 Bernstein polynomials for trajectory generation

Observability performance index For keeping high the excitation level of trajectories generated by multiple UAVs while reducing the complexity of the trajectories themselves, a support is given by the *obsevability performance index* $\underline{\sigma}(s(t), m)$. This index is defined in [2] as the minimum singular value of $\mathcal{O}(s(t), m) := \frac{1}{m} \sum_{i=s(t)-m+1}^{s(t)} H(\tau_i) H^T(\tau_i)$, which is used for determine the *Persistently Excitation* condition for the H matrix [11].
Exploiting the observability performance index variations, it is possible to note that $\underline{\sigma}(s(t), m)$ increases as we increase whether the trajectory complexity or the number of UAVs. Therefore, $\underline{\sigma}$ can be used as an additional parameter for the formulation of the trajectory generation problem.

Bernstein Polynomials The trajectory generation for multi-robot tasks can be done by exploiting the particular geometric properties of **Bernstein polynomials**; in particular, it is possible to employ direct methods using Bernstein polynomials approximation [2] for exploiting optimal control problems as finite-dimensional optimization problems [6] The generic *N-th order Bernstein polynomial* $p_{N,k}(t)$, defined in the equation 17, approximates $p_{d,k} : [t_0, t_f] \to \mathbb{R}^3$, which represents the *desired trajectory* to be tracked by the *k-th* UAV

$$p_{N,k}(t) = \sum_{j=0}^{N} \bar{p}_{j,N}^{[k]} b_{j,N}(t) \qquad t \in [t_0, t_f] \tag{17}$$

where

- $\bar{p}_{0,N}^{[k]}, ..., \bar{p}_{N,N}^{[k]} \in \mathbb{R}^3$ represents *Bernstein polynomial coefficients*
- $b_{j,N}(t) = \binom{N}{j} \frac{(t-t_0)^j (t_f - t)^{N-j}}{(t_f - t_0)^N}$ defines the *N-th order Bernstein basis*

Trajectory generation problem In the context of S&R missions, it is fundamental to generate favourable trajectories $p_{d,k}(t)$ with $k = 1, ..., n$ for the n ARTVA receivers for performing an accurate estimate of the location of the single ARTVA transmitter.
Assuming that the drones are capable to completely track their desired trajectories, the task is to find both the desired trajectories $p_{d,k}^*$ and the minimal final mission time t_f^*, aiming to minimize the actuation effort and maximize the observability index $\underline{\sigma}$ [2].

The Bernstein coefficients $\bar{p}_{j,k} = col(\bar{p}_{j,N}^{[0]}, ..., \bar{p}_{j,N}^{[n]}) \in \mathbb{R}^{3n}$ can be exploited for describing the n UAVs trajectories (for all $j = 0, ..., N$); the problem can be formulated as a nonlinear programming one, solvable using nonlinear optimization solver. Thus, the trajectory generation problem can be stated as follows

$$\min_{\bar{p}_{0,N}, ..., \bar{p}_{N,N}, t_f} w_1(t_f - t_0) + w_2 \sum_{k=1}^{n} \int_{t_0}^{tf} ||\ddot{p}_{N,k}(\tau)||^2 d\tau - w_3 \underline{\sigma}(s(t_f), s(t_f) - s(t_0)) \tag{18}$$

[11] For more details see [2]

subject to

$$\begin{aligned} p_{N,k}(t_0) &= p_{k,t_0} \\ \dot{p}_{N,k}(t_0) &= \dot{p}_{k,t_0} \end{aligned} \tag{19}$$

$$||p_{N,k}(t_f) - \hat{p}_t(\tau_{s(t_0)})|| \leq \delta_t \tag{20}$$

$$||\dot{p}_{N,k}(t)|| \leq v_{k,max}^2, \quad t \in [t_0, t_f] \tag{21}$$

for all $k, j = 1, ..., n, k \neq j$, with $w_1, w_2, w_3 > 0$.

The first conditions in 19 guarantees the continuity of the reference and its first derivative at time t_0. The constraint in 20 ensures the arrival of the receiver at a δ_t -neighborhood of the transmitter estimated position, by changing the design vale δ_t. The last condition guarantees that the reference trajectories can be tracked by the receivers. More in details, the positive constant $v_{k,max}$ in 21 define maximum allowed speed

3 Implementation

Our project is focused on the implementation of a Search and Rescue mission simulated on MATLAB, similar to the solution illustrated in [2]. The innovations developed with respect to the cited paper are the following:

- Instead of implement just a centralized system, we provide a solution exploiting a **decentralized control mode** and comparing it with the centralized one
- A **consensus protocol** is added to the system for managing the sharing of information between the agents; moreover, the trajectories followed by the drones are based on a PD controller on the current estimate
- The overall estimate of the target location is obtained on the basis of the individual estimations computed singularly from the data collected by each agent
- The framework developed supports whether 2D or 3D settings

In this section, we will take a closer look to all the implementation choices taken during the project, which have led us to the solution adopted.

3.1 Framework

3.1.1 ARTVA modeling - *getARTVAsig.m*

The overall work starts from the implementation of the mathematical models of the ARTVA system for a single UAV, and then passing through the development of a multi-veichles system in order to estimate the transmitter location by exploiting the information coming from all the drones in the formation.
More in details, inside "*utils*" folder:

- *getARTVAsig.m* implements a function computing the magnetic field read by the receiver 6, which is obtained from the mathematical model of electromagnetic vector field emitted by the ARTVA transmitter 3.
- In *eq8.m* there is the approximation 8 adopted for reducing the complexity of ARTVA model 6

3.1.2 Estimate of transmitter location - *RLS* folder

For the estimate of the transmitter position, we have exploited the Recursive Least Squares as the authors of [2]. The results of this algorithm are then used in 16 for computing the transmitter location estimate. The functions used for this purpose are located inside "*RLS*" folder, in which:

- *Estimate.m* contains *RLSStep* function, which firstly reconstructs the model 13 and then computes the RLS recursive step (namely, 14 and 15). *RLSStep* updates the estimate of the unknown vector whenever a new ARTVA sample is received.
- In *buildPhi.m* there is the function building the vector Φ 11 used in *Estimate.m*, for whether 3D or 2D positions
- *extractTarget.m* exploits the estimate of the unknown vector computed in *Estimate.m* for computing the inverse mapping 16 for extracting the target estimate.

3.1.3 Trajectory planning - *planningProblem.m*

For obtaining an accurate estimate of the location of the ARTVA transmitter, we have generated trajectories for the UAVs exploiting the Bernstein coefficients, as described in 2.2.5. In *planningProblem.m* we have built the simplified planning problem 18 using Bernstein approximants, subject to the boundary conditions 19, the end point constraint 20 and the inequality 21. The Bernstein polynomials approximants are applied by using the library inside *bernstein* folder

3.1.4 Centralized modeling - *Environment.m*

For comparing the decentralized system described in the next section, we have implemented a centralized environment class (*Environment.m*), which uses all the already mentioned MATLAB implemented resources inside its functions (namely, *updateEstimate* and *planTrajectories*), in order to get estimates of the transmitter location and generate new trajectories every time it is needed.

The UAVs are represented through *Agent.m* class. Each agent is characterized by:

- dynamics equations, implemented as a **double integrator** in *doubleInt.m* class, located in *model* folder
- a reference trajectory, that we have specified as Bernstein polynomials through *updateReference* function
- a controller following a Bernstein polynomial trajectory through PD and feedforward, that is implemented in *PDffwBrt* function

The trajectories replanning is regulated by *eventsFcn* function, looking for two kinds of events: the periodic arrival of a new ARTVA sample or the satisfaction of replanning conditions. *f* function provides the dynamic equations of all the agents. Finally, *sim* function generates a plot of agents positions in time.

3.1.5 Consensus protocol - *decentralized folder*

The implementation of the decentralized controller begins from the definition of the undirected graph representing the UAVs formation. In our project we choose to implement an "pentagonal network", namely composed by 5 drones equipped by ARTVA receivers which are connected in such a way to form a pentagon, as in 1. The adjacency matrix $A \in \mathbb{R}^{5x5}$ for this graph is the following:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix} \tag{22}$$

The degree matrix and the Laplacian matrix are computed in *adj2Laplacian.m* and then the overall formation plot is implemented in *vizFormation.m*.

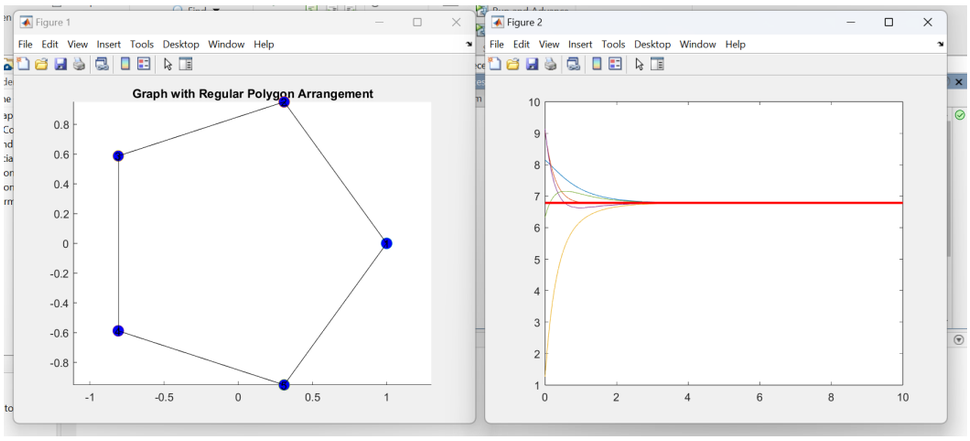

Figure 3: The picture on the left shows the undirected graph representing the formation of the UAVs network; the picture on the right illustrate the convergence of the agents to the ground truth (red line)

3.1.6 Decentralized modeling - *EnvironmentDec.m*

The definition of the decentralized enviroment class starts by defining a connection for the agents, which is the one seen in 3.1.5, by exploiting *circleConnection.m*, *adj2laplacian.m* and *expandedLaplacian.m*. The dynamic equations of all the agents and the consensus dynamics are implemented in f function.
The *AgentDec.m* contains a class for describing the agents quite similar to the one in *Agent.m*. The difference is that each decentralized agent keeps its own estimate and then updates the estimate of target using both its own information and the neighbors data. In *EnvironmentDec.m* each agent executes its own routines related both to RLS and to the trajectorie planning; after that, they merge the information coming from each of them through the consensus protocol, refining the convergence to the actual estimate of the transmitter location. Finally,the MATLAB file implements two figures: the first one shows the positions of the agents in time converging to the estimated position of the target; while the second one plot the consensus evolution with respect to the coordinates (x,y for 2D and x,y,z for 3D setting) of the position of the transmitter 4.

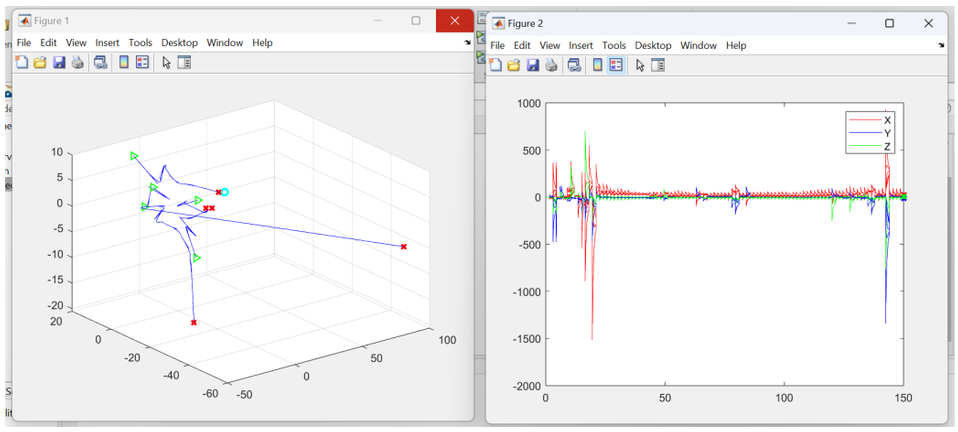

Figure 4: The picture on the left shows the initial position (green triangles) and the final position (red crosses) of the agents, with the tracking of the evolution of drones position in time, trying to reach the actual transmitter position (light blue circle); the picture on the right, illustrates the consensus evolution of X,Y,Z coordinates for each agent

3.1.7 Test files

The following MATLAB files are used for testing the overall implemented system, providing also graphical outcomes as simulations. In particular:

- *test.m* refers to the centralized model-based system
- *testDec.m* refers to the decentralized model-based system
- *testConsensus.m* provides graphical results related to the consensus protocol imposed on the UAVs network, plotting the ground truth convergence value

4 Simulations

Simulations have been carried out in order to assess the correct implementation of the methodologies and to evaluate the results. Several test have been run, both for centralized and decentralized architectures. For the latter, has been tried different combinations of number of agents, which went from 4 to 6, formations of the agents, and 2D or 3D simulation scenario, as show respectively in the illustrations 5, 7 for 2D cases and 6, 8 for the 3D ones. In particular, in each figure represents on the left the plot of trajectories performed by the agents, ending at the red crosses, and in which the blue circle is the target; on the right, there is the consensus evolution of X and Y (and also Z in 3D case), where the colored crosses are the real values target position.

Regarding the formations, they influences which agents are connected together. Two different possibilities have been explored: one named *"Circle"* in which the connections of the graph that represents the agent create a polygon shape as in 4 , i.e. each agent is connected only with 2 agents, the previous and the next one along the perimeter of the circle. The second one, named *"Star"*, has one agent in the center and all the other are connected only with it. For what regard the centralized model, has been tested 2D and 3D scenarios with 4, 5 or 6 agents. Time needed to perform each simulation is in the order of some seconds, so it's not relevant to report them. Eventually, 10 runs are performed for each combination of features and then the average error has been computed for the estimation of both target and distance from the target itself.

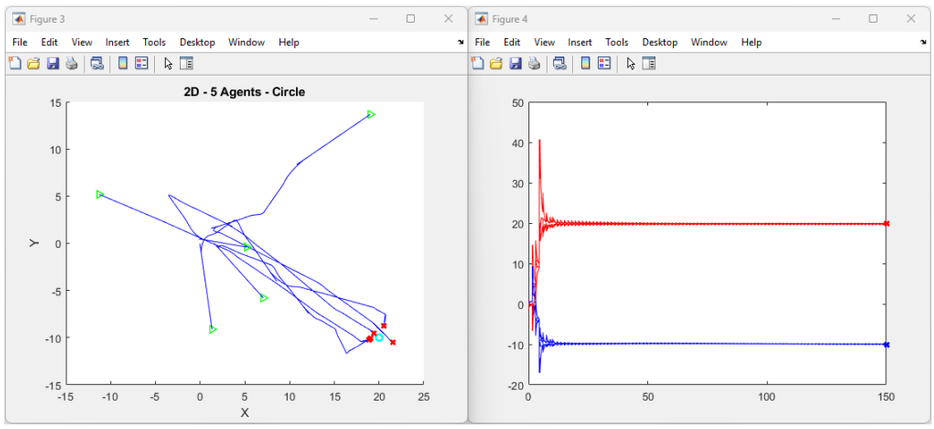

Figure 5: 5 agents in a 2D decentralized environment in 'Circle' formation.

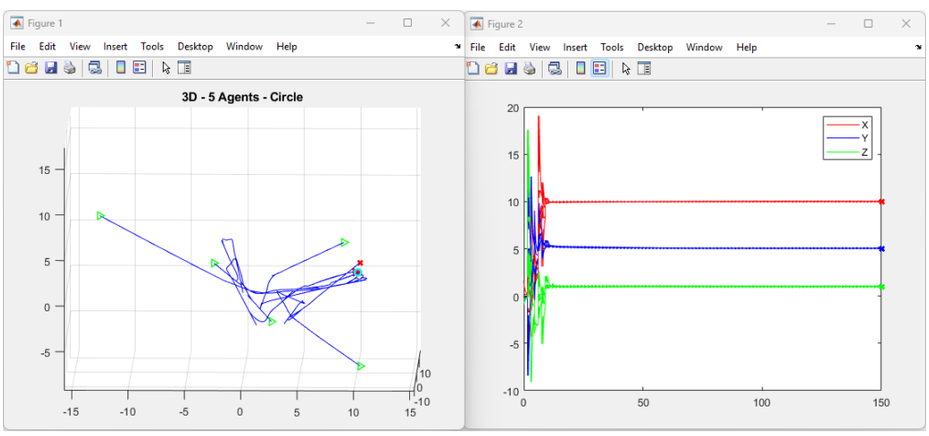

Figure 6: 5 agents in a 3D decentralized environment in 'Circle' formation.

5 Results

As can be seen from the images 5,6,7 and 8, results tend to differ between the two tested formations, 'Circle' and 'Star'. In 'Star' formation, the consensus evolution of the estimate of X,Y and Z (in the 3D scenario) fluctuates a lot, and furthermore in the 3D case, as can be seen in 8, not all agents correctly estimate the target coordinates. While in 'Circle' formation in 5 and 6 X,Y and Z evolutions are more precise, tending also to estimate the correct values in much less time and remaining stable in their guess once they have reached them. For what regard the trajectories of the agents, in both formations they look pretty similar , except for the fact that the final positions are closer to the target in 'Circle' formations than in 'Star', resulting in a smaller final error.

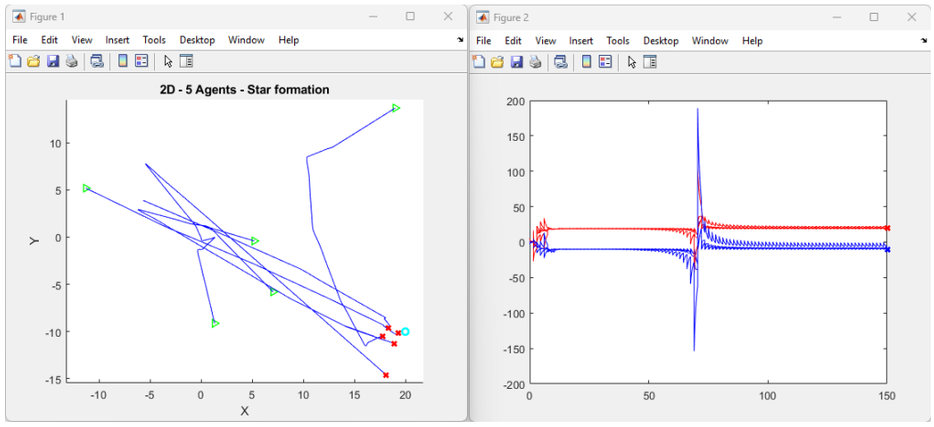

Figure 7: 5 agents in a 2D decentralized environment in 'Star' formation.

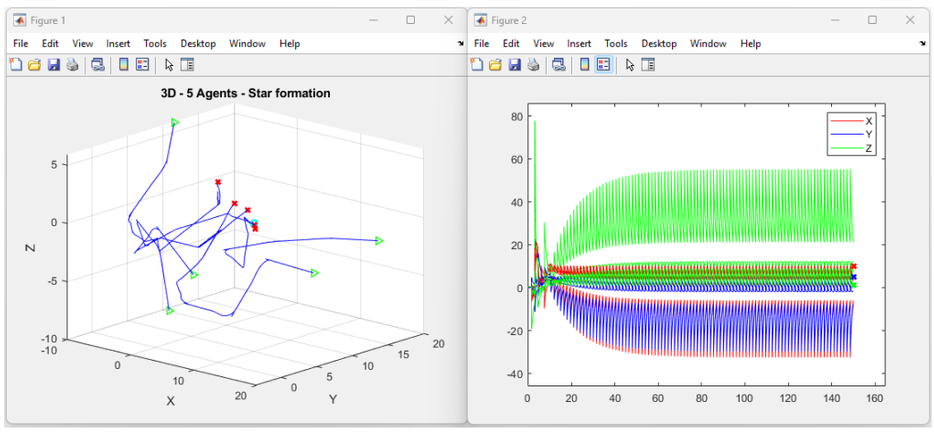

Figure 8: 5 agents in a 3D decentralized environment in 'Star' formation.

This can be seen also from the error results reported in 1: the error is the mean value averaged over 10 run for each setting and over the X,Y and Z variables. 'Star' environment perform worse than 'Circle', both in 2D and 3D case. In general 2D simulations return a lower error with respect the 3D ones. On the other hand,centralized method's error has an order of magnitude less than decentralized mean errors. Visualization of centralized implementation results are report in 9 and 10

Environment	Dimension	Mean Error
Star	2D	0.589
Star	3D	0.717
Circle	2D	0.127
Circle	3D	0.189
Centralized	2D	0.013
Centralized	3D	0.017

Table 1: Results table with error values averaged over 10 runs

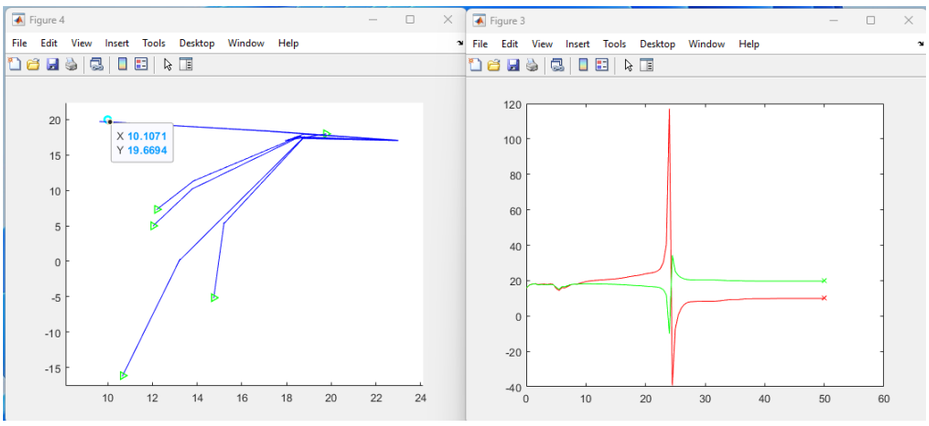

Figure 9: 5 agents in a 2D environment.

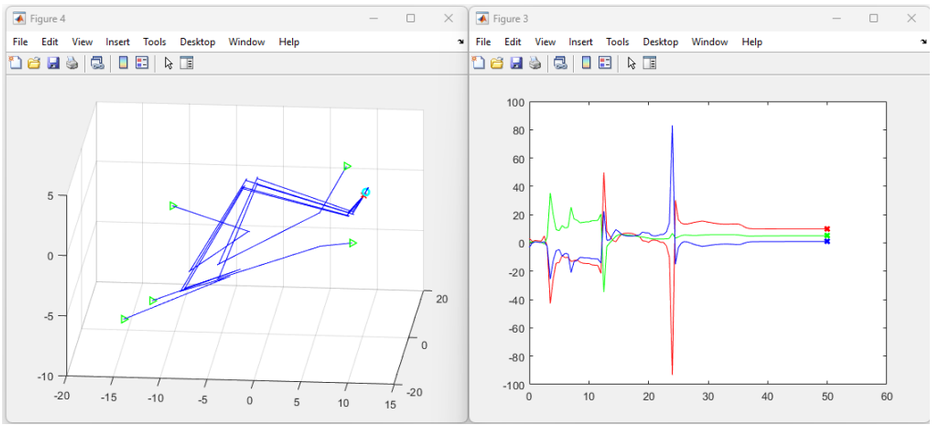

Figure 10: 5 agents in a 3D centralized environment.

6 Conclusions

In this document we have shown how to conduct a Search and Rescue mission with multiple UAVs carrying an ARTVA receiver. Using multiple agents provides serious benefits in terms of total mission time and energy requirements (per robot). The proposed approach is a decentralized extension of [2], where each agent only has access to partial information of itself and its neighbors. The advantages of using a decentralized approach is that there is no need for a central authority that performs trajectory planning as each agent is equipped to perform its own planning, moreover while the cumulative computation is higher, it is spread equally on each agent which results in less performance requirements. Lastly the approach scales seamlessly to any number of agents, addition and removing of agents and connections without modifying the algorithm. On the other hand, it brings certain disadvantages compared to a centralized approach. First of all, the estimation of the target might be poorer, because each agent does not use every information that is available. The decentralized method tries to compensate for this with a consensus protocol through which the formation converges to a single estimate. The simulations conducted in two and three dimensions have shown that while the decentralized method correctly estimates the target and makes the agent move close to the correct position, and while it comes at the cost of slightly increased mission time, the whole performance is still somewhat accurate and within reasonable accuracy limits in order for it to be usable in a Search and Rescue mission. The topology of the connection is shown to be important but not crucial to the algorithm's success, as long as the graph is connected (in order for consensus to converge).

References

[1] L. Marconi, C. Melchiorri, M. Beetz, D. Pangercic, R. Siegwart, S. Leutenegger, R. Carloni, S. Stramigioli, H. Bruyninckx, P. Doherty, *et al.*, "The sherpa project: Smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments," in *2012 IEEE international symposium on safety, security, and rescue robotics (SSRR)*, pp. 1–4, IEEE, 2012.

[2] C. Tabasso, N. Mimmo, V. Cichella, and L. Marconi, "Optimal motion planning for localization of avalanche victims by multiple uavs," *IEEE control systems letters*, vol. 5, no. 6, pp. 2054–2059, 2021.

[3] P. Piniés and J. D. Tardós, "Fast localization of avalanche victims using sum of gaussians," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 3989–3994, IEEE, 2006.

[4] N. Mimmo, P. Bernard, and L. Marconi, "Avalanche victim search via robust observers," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 4, pp. 1450–1461, 2020.

[5] P. Ioannou and J. Sun, "Robust adaptive control. chelmsford, ma, usa: Courier corporation," *Chelmsford MA USA: Courier Corporation*, 2012.

[6] V. Cichella, I. Kaminer, C. Walton, N. Hovakimyan, and A. Pascoal, "Bernstein approximation of optimal control problems," *arXiv preprint arXiv:1812.06132*, 2018.