

ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA

SCHOOL OF ENGINEERING

Department of
Electrical, Electronic and Information Engineering
“Guglielmo Marconi”
DEI

Master’s Degree in Automation Engineering

Master Thesis
in
Distributed Autonomous Systems

**Distributed Persistent Monitoring of
Moving Targets Using High Order
Control Barrier Functions**

Candidate:

Lorenzo Balandi

Supervisor:

prof. Giuseppe Notarstefano

Co-supervisors:

Dr. Paolo Robuffo Giordano

Dr. Andrea Testa

Dr. Nicola De Carli

Academic Year

2021–2022

Session

3

Contents

Abstract	4
Introduction	5
Motivations	5
Literature	5
Organization	8
1 Modeling	9
1.1 Distributed Network	9
1.2 Target Model	10
1.3 Alternative Target Model	11
1.4 Measurement Model	11
1.4.1 Practical implementation considerations	13
2 Estimation	15
2.1 Information Consensus Filter	15
2.2 ICF for the Alternative Target Model	19
2.3 ICF Stability	20
3 Target Monitoring	22
3.1 Information Measure	22
3.2 Perception Awareness	24
3.3 High Order Control Barrier Functions	25
3.4 Persistent Target Monitoring	28
3.4.1 Centralized Problem	28
3.4.2 Distributed Problem	30
4 Simulations	35
4.1 Case 1: Single target with linear trajectory and no additional tasks	37
4.2 Case 2: Multi-target with sinusoidal trajectory and no addi- tional tasks	41

4.3 Case 3: Single target with linear trajectory and bearing-only formation control	42
5 Conclusions and Future Works	50
A Matrix calculus rules	51
B Computations	53
B.1 Computation of $L_f(h_r(\mathbf{x}))$	54
B.2 Computation of $L_{g_i}\psi_{1r}(\mathbf{x})$	54
B.3 Computation of $L_f\psi_{1r}(\mathbf{x})$	56
Acknowledgements	59
Bibliography	63

Abstract

This Thesis considers the problem of persistently monitoring a set of moving targets using a team of aerial vehicles. Each agent of the network is assumed equipped with a camera providing bearing measurements with a limited FoV, and it implements an Information Consensus Filter (ICF) to estimate the state of the target(s). It is shown that, thanks to a suitable choice of the output equation, the filter can be proven to be globally exponentially convergent under *Persistency of Excitation* (PE) conditions. This is then leveraged for proposing a distributed control scheme that allows maintaining a prescribed minimum PE level for ensuring filter convergence. At the same time, the agents in the group are also allowed to perform additional tasks of interest while maintaining a collective observability of the target(s). In order to enforce satisfaction of the observability constraint, two main tools are exploited: (i) the weighted Observability Gramian with a forgetting factor as a measure of the cumulative acquired information, and (ii) the use of High Order Control Barrier Functions (HOCBFs) as a mean to maintain a minimum level of observability for the targets. Simulation results are reported to prove the effectiveness of this approach.

Introduction

Motivations

Mobile robots are ubiquitous in modern society. They can be found in many industrial fields such as precision agriculture, inspection and surveillance and their use is rapidly expanding. The capabilities of a single robot are increased if the robot is part of a team, which can be seen as a distributed network of mobile sensors that can perceive and interact with the environment. Multi-robot systems are thus a topic of great interest, particularly if in the system there is no need to communicate with a centralized node because each robot can sense, compute and communicate by its own. From a scientific point of view, these systems are an example of the application of Network Systems theory to the world of Mobile Robotics, and they have therefore aroused great interest in the academic community since they unify many different topics and theories.

This Thesis explores the topics of distributed estimation and control of multi-robot systems, proposing a novel approach to the problem of Persistent Monitoring of moving targets leveraging Control Theory tools such as High Order Control Barrier Functions. It is thus part of this wide framework of high importance for both academic and industrial worlds.

Literature

Target tracking is a classical topic in the multi-robot community. In the classical formulation of the problem, a group of (possibly) mobile sensors needs to cooperatively track the position of one or more moving targets. Each mobile sensor can obtain a measurement of the target and fuse it with the other sensors in order to obtain a better estimate [1]–[3]. The sensor fusion has been tackled in many works, and some distributed filters have been proposed, such as the Kalman-Consensus Filter (KCF) [1] or the Information Consensus Filter [4] (ICF, used in this work in a modified version). Mobile sensors can also optimize their positioning/moving so as to maximize the information collected about the target state [5]–[9], thus

improving the localization accuracy. This field is often denoted as *active sensing*: the optimization of the sensor motion/placement usually relies on a suitable information metric, which is often a function of the eigenvalues of an information matrix, e.g., the well-known *Fisher Information Matrix* [10], the *Observability Gramian* (OG) [11], or, alternatively, the covariance matrix [5], [6], [12]. In particular, [5] proposes a fully distributed solution using *dynamic average consensus estimators* [13], [14] and a gradient-based controller to define the motion of the sensors. On the other hand, [6] proposes a centralized solution in which the problem of the self-localization of the sensors is also faced and unified with the target tracking.

The sensors considered in the target tracking works can be range-only, bearing-only, range-bearing or a combination of heterogeneous sensing devices. Target tracking implies a target-centered approach, hence all the control effort is used to pursue the tracking goal and it is difficult to perform other tasks.

In this Thesis, differently from many previous works on this subject, we consider a situation in which the target localization is not the only task for the robot group. Therefore, we only aim at maintaining a *minimum level* of accuracy for localizing the moving target(s) while the group redundancy can be exploited for realizing additional tasks of interest. A situation in which this strategy could be desirable is the case in which an heterogeneous group of robots composed of quadrotors and ground robots is collaborating. The quadrotors can be assumed able to localize themselves in a common frame (e.g. using GPS or by running a distributed localization algorithm such as [15]). The same, however, may not hold for the ground robots which typically have more limited sensing capabilities (or operate in a GPS-denied environment). In this case, the quadrotors can act as a moving localization system for the ground robots and exploit their group redundancy for optimizing their motion and achieving any other task of interest (e.g., formation control, coverage, and so on). Note, however, that the proposed algorithm is general and works also without the assumption of ground targets.

Persistent Monitoring, differently from target tracking, requires an area-centered approach where a team of robots have to visit an area of interest at a certain frequency [16]. The types of vehicles considered in these works are usually fixed-wing aircrafts, quadcopters or Micro Aerial Vehicles (MAVs). Many works have been proposed, initially extending solutions for single vehicles to multiple robots [17], [18]. Another approach to the persistent monitoring problem is to cast it as a vehicle routing problem [19], [20] in which the robots have to visit a discrete set of locations, often with time and/or energy constraints. However, most of these methods for high-level vehicle control consider either approaches with a formal proof of optimality but not scalable in the number of vehicles or decentralized and scalable approaches but heuristic and policy-based.

The approach proposed in this work differs from all others about persistent monitoring, since the problem is not based on policies, it is presented in the 3-dimensional case and the proposed algorithm can scale up to an arbitrary number of robots.

In this work, we propose a distributed algorithm based on *Control Barrier Functions* (CBFs). CBFs have emerged as a powerful tool to ensure constraints satisfaction while optimizing performance in nonlinear control problems [21]. Compared to *barrier Lyapunov functions*, which have a vertical asymptote when approaching the boundary of the allowed set [22], CBFs are well-defined also outside the *safe set* and can thus provide robustness to noise and asymptotic stability of the safe set [23]. When the constraint is imposed on an output, i.e. a function of the state whose first derivative does not directly depend on the inputs (i.e., its *relative degree* is higher than one), imposing invariance of the safe set becomes more involved. Some solutions have been proposed, such as *Exponential Control Barrier Functions* (ECBFs) [24] and, more recently, *High Order Control Barrier Functions* (HOCBFs) [25]–[27]. In this work, in particular, we exploit the HOCBFs presented in [27] due to their greater robustness to noise and their more general formulation. A brief overview on CBFs and HOCBFs can be found in Chapter 3. CBFs have also been employed for multi-agent systems, e.g., for collision avoidance [28], connectivity maintenance [29], and temporal logic tasks [30]. However, some of these works do not actually propose a fully distributed solution. Conditions to satisfy the centralized constraint by solving local QPs have been provided, for example, in [30]. In this work, we apply HOCBFs to a multi-agent system proposing a fully decentralized solution in which each agent of the network only uses local data and data from neighbors.

Contributions

The goal of in this work is to design a decentralized algorithm based on HOCBFs for a group of quadrotor UAVs that need to localize (multiple) moving targets. In particular, the ideas in [31] are extended to the multi-robot target localization case and make use of the Information Consensus Filter (ICF) [2] for estimating the target states. It is shown that, under mild assumptions, the ICF stability proof in [3] for a linear system is still valid in the case in analysis with a nonlinear measurement equation under suitable Persistency of Excitation (PE) conditions. It is then designed a distributed control for the multi-UAV system based on HOCBFs able to guarantee a minimum level of PE (necessary for the ICF convergence) while also allowing the execution of other tasks of interest. Indeed, thanks to the optimization framework required by the HOCBFs, an existing controller can be minimally modified to guarantee that some safety constraints are satisfied. Moreover,

it is possible to easily add other constraints to the optimization problem, which can be practically solved efficiently with widespread tools. The proposed approach allows to use the minimum number of drones to achieve the localization task. The application of HOCBFs to target localization is a novelty in the distributed estimation literature and presents many advantages compared to the classical solutions in which a unique control objective can be pursued.

Organization

This Thesis is organized as follows. The modeling part is presented first, in order to define the terminology and the variables that will be used in the continuation of the work. The agents and targets models are explained, and an alternative model for the targets is provided. Particular emphasis is given to the measurement model, where the problem of a real implementation is also discussed.

The estimation chapter is then presented. The modifications made to the Information Consensus Filter to work with bearing measurements are discussed and its stability under some conditions is proved.

The next chapter present the main contribution of this work. It first provides some basic information about the measure used to represent the information, the method used to make the drones aware of their perception limits and the Control Barrier Functions (CBFs) in their first order and higher order case. These concepts are then used to explain how the persistent monitoring of multiple targets is achieved.

Subsequently, the simulation results are reported and discussed to show the effectiveness of the proposed approach.

Finally, after the conclusions, two appendices show in detail the computations introduced in the previous chapters.

Chapter 1

Modeling

This chapter provides an explanation of the distributed network consisting of the drones and reports the model assumed for them, aside from the model used to represent the targets and finally the measurement model through which the quadrotors sense the targets. An alternative model for the targets and some practical considerations about the sensing model are also reported. The concepts presented in this Chapter will be extensively used in the continuation of the Thesis.

1.1 Distributed Network

Consider N quadrotors, which sense M target robots. The quadrotors are assumed to be localized in a common frame, while the targets need to be localized by the quadrotors using relative measurements. The targets are relatively slowly moving and they could be, for example, ground robots, which have no possibilities to autonomously localize themselves. Assume that N and M are known by all agents of the network (anyway, if N was not known, it could be estimated [32]).

The drones form a distributed system in which each agent can communicate only with its neighbors according to a fixed, undirected and connected communication graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where \mathcal{V} is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. The set of neighbors of the i -th robot is denoted as $\mathcal{N}_i \triangleq \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$. Since the communication is bidirectional $(i, j) \in \mathcal{E} \iff (j, i) \in \mathcal{E}$. No communication delay or data loss is considered.

The quadrotors are modelled as single integrators, with the state of agent i being its position $\mathbf{p}_i \in \mathbb{R}^3$:

$$\dot{\mathbf{p}}_i = \mathbf{u}_i, \quad i = 1, \dots, N \quad (1.1)$$

where $\mathbf{u}_i \in \mathbb{R}^3$ is its velocity input.

A representation of positions, communication and sensing of agents and targets is depicted in Figure 1.1.

It is useful to define the aggregate drones positions

$$\mathbf{p} = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_N \end{bmatrix} \in \mathbb{R}^{3N}$$

and analogously the aggregate drones inputs

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_N \end{bmatrix} \in \mathbb{R}^{3N}$$

that will be used in the next chapters.

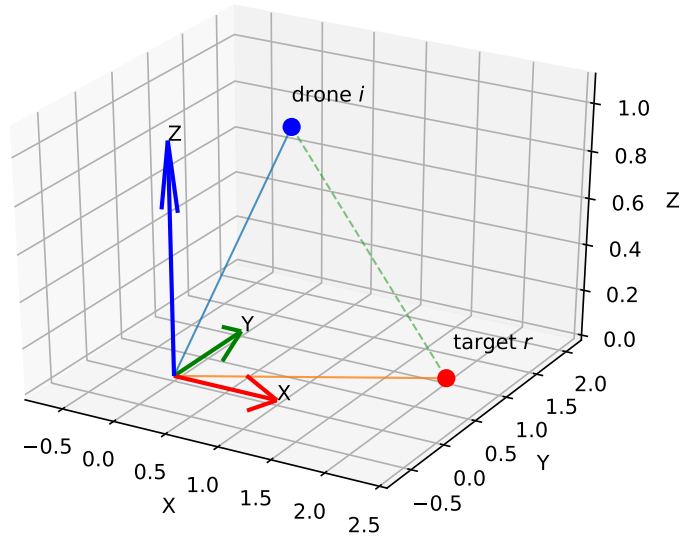


Figure 1.1: Positions of drone i and target r in the global reference frame.

1.2 Target Model

The model that the drones consider for the target for estimation purposes is a double integrator with constant zero acceleration, namely a constant

velocity model. In this way, both positions and velocities of the targets are estimated. The state considered for the r -th target is

$$\mathbf{x}_r^\tau = \begin{bmatrix} \mathbf{p}_r^\tau \\ \mathbf{v}_r^\tau \end{bmatrix} \in \mathbb{R}^6 \quad (1.2)$$

where \mathbf{p}_r^τ is the position and \mathbf{v}_r^τ is the velocity. The state dynamics is

$$\dot{\mathbf{x}}_r^\tau = \begin{bmatrix} \dot{\mathbf{p}}_r^\tau \\ \dot{\mathbf{v}}_r^\tau \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{p}_r^\tau \\ \mathbf{v}_r^\tau \end{bmatrix} = \begin{bmatrix} \mathbf{v}_r^\tau \\ \mathbf{0} \end{bmatrix} \quad r = 1, \dots, M \quad (1.3)$$

Since the assumed velocity for the targets is constant, an acceleration different from zero will be considered a disturbance by the quadcopters.

For control purposes, the targets are instead modeled as single integrators with control input \mathbf{v}_r^τ

$$\dot{\mathbf{p}}_r^\tau = \mathbf{v}_r^\tau \quad r = 1, \dots, M \quad (1.4)$$

The reason of this distinction is that the proposed distributed control algorithm will only need knowledge of the (estimated) target velocity.

1.3 Alternative Target Model

It is also possible, for estimation purposes, to model the targets as single integrators

$$\dot{\mathbf{p}}_r^\tau = \mathbf{u}_r^\tau, \quad r = 1, \dots, M \quad (1.5)$$

with velocity input \mathbf{u}_r^τ and state given only by the position \mathbf{p}_r^τ . With this model, it is assumed that the targets communicate their velocities to the drones. As mentioned before, the targets could be, for example, collaborating ground robots, which have no possibilities to autonomously localize themselves. The drones can thus serve as a mobile localization system for the robots. In this case, the communication is bidirectional (drones to targets and targets to drones). The reason why the velocity of each target must be communicated to the drones is that, as mentioned before, it is needed to prove the convergence of the filter discussed in the next chapter, and it can be either estimated using a double integrator model or directly communicated using this model.

1.4 Measurement Model

Each drone is embedded with an onboard camera pointing downside from which it acquires relative bearing measurements $\beta_{ir} \in \mathbb{S}^2$ with respect to the target(s):

$$\beta_{ir} = \frac{\mathbf{p}_{ir}}{d_{ir}} \quad (1.6)$$

where $\mathbf{p}_{ir} = \mathbf{p}_r^\tau - \mathbf{p}_i$ and $d_{ir} = \|\mathbf{p}_r^\tau - \mathbf{p}_i\|$. Note that, since $\|\boldsymbol{\beta}_{ir}\| = 1$, the three components of the bearing are not independent.

A bearing measurement $\boldsymbol{\beta}_{ir}$ with respect to the r -th target is acquired if the target is in a certain range w.r.t. the i -th drone, i.e. $D_m < d_{ir} < D_M$ and the target is in the FoV of the drone, i.e. $-\bar{x}_M \leq \bar{x} \leq \bar{x}_M$ and $-\bar{y}_M \leq \bar{y} \leq \bar{y}_M$, where \bar{x} and \bar{y} are the image plane coordinates of the target and \bar{x}_M and \bar{y}_M the corresponding FoV limits. A graphical representation of the FoV is depicted in Figure 1.2. It is assumed that the camera frame has origin coincident with the frame attached to drone i and opposite z axis.

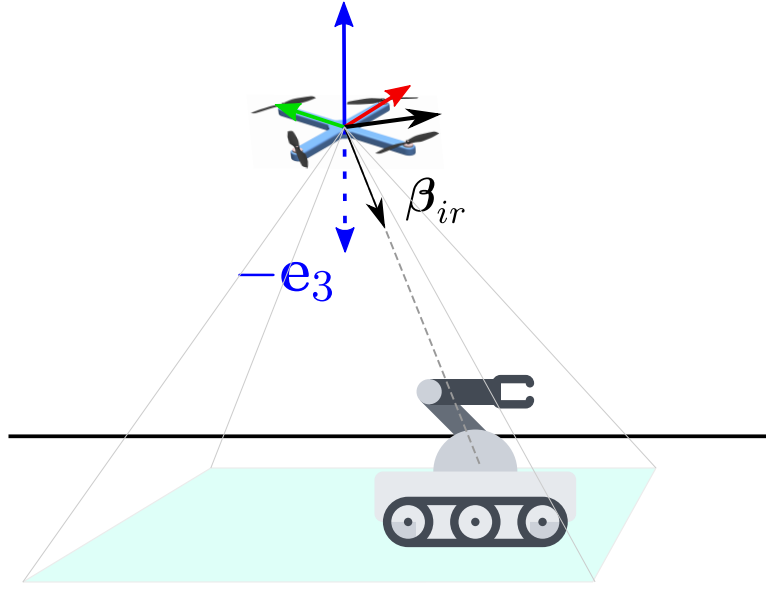


Figure 1.2: Graphical representation of the FoV limitation and the relative bearing measurement. The optical axis is aligned with axis $-z$ (blue) of the reference frame attached to drone i .

In case of noisy measurements, the noise enters as follows

$$\boldsymbol{\beta}_{ir} = \frac{\bar{\mathbf{p}}_{ir} + \boldsymbol{\nu}_i}{\|\bar{\mathbf{p}}_{ir} + \boldsymbol{\nu}_i\|} \quad (1.7)$$

with

$$\bar{\mathbf{p}}_{ir} = \frac{\mathbf{p}_{ir}}{-(\mathbf{p}_{ir})_z} \quad (1.8)$$

which is highly nonlinear. Basically we divide \mathbf{p}_{ir} by the opposite of the z -coordinate in order to obtain -1 as third component of the vector (to express it in the drone coordinates) and we add the measurement noise $\boldsymbol{\nu}_i$,

assumed Gaussian distributed in x and y and null in z

$$\boldsymbol{\nu}_i = \begin{bmatrix} x\nu_i \\ y\nu_i \\ 0 \end{bmatrix} \quad (1.9)$$

This simulate the measure that we can obtain from an image, assuming that the camera is calibrated, thus giving a measure of type $[x, y, -1]^T$, where the -1 is due to the downward orientation of the camera, whose z axis has opposite direction of the corresponding axis of the frame attached to the drone. Note that, even if the noise is Gaussian in x and y , it is subject to nonlinear transformations and thus the error that affects the final measure used in the filter will not be Gaussian. Indeed, the noise is Gaussian only at image level, but from the image we derive a bearing which is then used to build the projection matrix used for the estimator.

1.4.1 Practical implementation considerations

In a practical implementation, we would apply to the image obtained from a calibrated camera an image detection algorithm such as [33] or simpler computer vision algorithms, which usually give one or more bounding boxes as output. Considering the center of the bounding box, we obtain the pixel coordinates $[u, v]^T$, which are then expressed in image plane coordinates $[x_p, y_p, 1]^T$, which are finally expressed in the drone coordinates considering the relative position and orientation of the camera with respect to the drone reference frame, namely the reference frame attached to the drone. The passage from the pixel coordinates to the image plane coordinates requires the knowledge of parameters obtained with camera calibration, hence the assumption of the calibrated camera. The scheme of these transformations is depicted in Figure 1.3. The transformed point in drone coordinates corresponds to the vector $\bar{\mathbf{p}}_{ir}$, which is used to compute the bearing as

$$\beta_{ir} = \frac{\bar{\mathbf{p}}_{ir}}{\|\bar{\mathbf{p}}_{ir}\|} \quad (1.10)$$

In the case under analysis in this work, it is sufficient to consider the image plane coordinates $[x_p, y_p, -1]^T$, where again the -1 is due to the downward orientation of the camera axis, because the image plane coordinates are expressed in the frame of the drone. As mentioned before, we are assuming that the camera frame has same origin of the i -th drone frame and opposite z axis, hence no other transformations are needed. In a real implementation, one should consider the rototranslation between the camera frame and the drone frame, in order to express the data obtained from the camera in the reference frame used for the rest of the algorithm.

Since the bounding boxes obtained from the computer vision algorithms

could be very discontinuous, it may be necessary to filter them in order to obtain a smoothly varying bearing and to reject outliers. An example of this can be found in [34], where a Kalman Filter is applied to the dimensions of the bounding box to give it a smooth dynamics.

Note that, in this work, we do not consider the roll, pitch and yaw angles of a real quadrotor, hence the FoV is not subject to variations due to accelerations. In a real case, this could be represented by a drone with a camera mounted on a 3-axis gimbal pointing downward.

With a view to practical implementation, the usage of a camera as the only exteroceptive sensor is reasonable due to its flexibility and (potentially) low cost, which have made this device widely used in robotics in recent years.

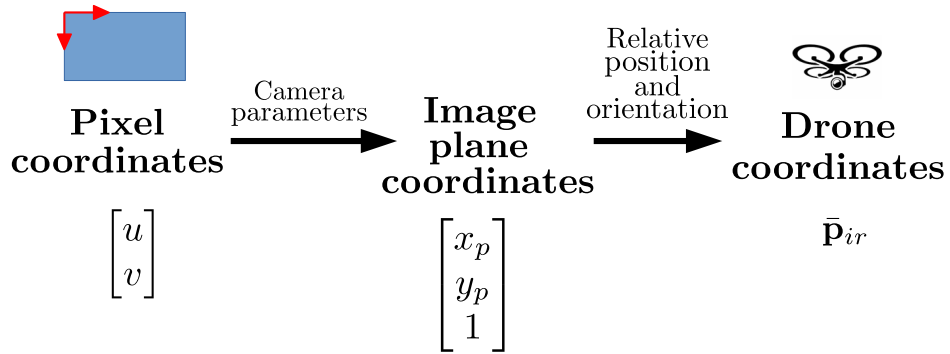


Figure 1.3: Scheme of the passages to transform pixel coordinates in drone coordinates.

Chapter 2

Estimation

In this chapter, the estimation scheme run by the drones to estimate the state of the target(s) is explained. An existing distributed information filter is modified to work with relative bearing measurements and with multiple targets, and the algorithm steps are briefly discussed. It is shown that linearizing the output equation by output injection it is possible to obtain a globally exponentially stable observer under suitable Persistency of Excitation (PE) conditions.

In the following, the subscript ir denotes a the local copy of a variable kept by drone $i \in \{1, \dots, N\}$ related to the target $r \in \{1, \dots, M\}$.

2.1 Information Consensus Filter

The drones estimate the state of the targets using a modified version of the Information Consensus Filter (ICF) presented in [4] and further discussed in [2]. Using this filter, only collective observability of the target(s) is required [3], i.e., it does not assume each robot to observe the target, as it is instead the case, for example, of the KCF discussed in [1]. The ICF has been proven to perform better than other distributed filters in distributed camera networks applications [2], especially when some nodes of the network have no direct measure of the target(s). The problem tackled in this Thesis can be seen as a distributed camera network in which each camera can move to perform some tasks of interest.

In each discrete time step k , the ICF includes K iterations to reach the consensus with the neighbors about the current estimation and the associated information matrix. Thanks to this consensus, also a drone without the direct measure of a target can receive the related estimate from its neighbors.

The *average consensus algorithm* [35] used in the filter is in the form:

$$a_i^k \leftarrow a_i^{k-1} + \epsilon \sum_{j \in \mathcal{N}_i} (a_j^{k-1} - a_i^{k-1}) \quad (2.1)$$

where a_i is the state (scalar or vectorial) of the i -th agent of a network of N agents. The update will make all the states converge to the average of the initial states. In (2.1), \mathcal{N}_i is the neighbor set of agent i and ϵ is the convergence rate parameter. To have stability $0 < \epsilon < \frac{1}{\Delta_{max}}$ must hold, where Δ_{max} is the maximum degree of the network (i.e. the maximum number of neighbors that an agent has in the network).

For the filter design we consider the discrete-time model of system (1.3) with constant velocity:

$$\mathbf{x}_r^\tau(k+1) = \mathbf{A}_d \mathbf{x}_r^\tau(k) + \gamma(k) \quad r = 1, \dots, M \quad (2.2)$$

where $\mathbf{A}_d = \begin{bmatrix} \mathbf{I} & \Delta T \mathbf{I} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}$, with ΔT being the discretization step, while $\gamma(k) \in \mathcal{N}(\mathbf{0}, \mathbf{Q})$ being gaussian process noise with positive definite covariance matrix \mathbf{Q} .

The observer does not directly uses the expression (1.6) as output model, instead, as in other works [31], it is considered an output equation linearized through output injection:

$$\mathbf{z}_{ir}(k) = \mathbf{\Pi}_{\beta_{ir}}(k) \mathbf{p}_r^\tau(k) \quad (2.3)$$

where

$$\mathbf{\Pi}_{\beta_{ir}} = \mathbf{I} - \beta_{ir} \beta_{ir}^T \in \mathbb{R}^{3 \times 3}$$

is an orthogonal projector. Note that, although (2.3) is nonlinear, it only depends on a *measured* (nonlinear) function of the state (the bearing β_{ir}). $\mathbf{\Pi}_{\beta_{ir}}$ satisfies the following properties:

- idempotence: $\mathbf{\Pi}_{\beta_{ir}} \mathbf{\Pi}_{\beta_{ir}} = \mathbf{\Pi}_{\beta_{ir}}^2 = \mathbf{\Pi}_{\beta_{ir}}$
- symmetry: $\mathbf{\Pi}_{\beta_{ir}} = \mathbf{\Pi}_{\beta_{ir}}^T$

These properties will be used in the following. The orthogonal projection matrix $\mathbf{\Pi}_{\beta_{ir}}$ geometrically projects any vector onto the orthogonal complement of β_{ir} . As a consequence, is it easy to verify that

$$\mathbf{\Pi}_{\beta_{ir}}(k) (\mathbf{p}_r^\tau(k) - \mathbf{p}_i(k)) = 0$$

As in [31], the information \mathbf{B}_{ir} associated to each measurement is simply taken as $\mathbf{B}_{ir} = b \mathbf{I}$, with $b > 0$. This can be done as from the perspective of the filter stability, this matrix is simply required to be a positive semi-definite gain matrix.

The steps of the modified ICF are reported in Algorithm 1 and briefly commented. For more details, however, the reader is referred to [2], [4]. Note that, in Algorithm 1, the notation of the original paper [4], which holds for a single target case, is adapted to the multiple targets case. Moreover, the symbol \dagger has been used to indicate the pseudo-inverse of a matrix, while the exponent $'-'$ is used to indicate the prior estimate and $'+'$ to indicate the posterior estimate.

To use the linear time-varying form given by the expression (2.3) in the ICF, a couple of algebraic passages are necessary. First, consider that by using the orthogonal projector properties, it follows:

$$\begin{aligned} \mathbf{\Pi}_{\beta_{ir}}(k) (\mathbf{p}_r^\tau(k) - \mathbf{p}_i(k)) = 0 &\implies \\ \implies \mathbf{z}_{ir}(k) = \mathbf{\Pi}_{\beta_{ir}}(k) \mathbf{p}_r^\tau(k) = \mathbf{\Pi}_{\beta_{ir}}(k) \mathbf{p}_i(k) \end{aligned} \quad (2.12)$$

Define the *observation matrix* as

$$\mathbf{H}_{ir} = \begin{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{3 \times 6} \quad (2.13)$$

where the matrix of zeros $\mathbf{0} \in \mathbb{R}^{3 \times 3}$ is necessary because through a bearing we can only directly gather information about the position of the target and not about its velocity (which is estimated using the filter).

The update passage from the original ICF [2] is modified as:

$$\begin{aligned} \mathbf{v}_{ir}^0 &= \frac{1}{N} \mathbf{W}_{ir}^-(k) \hat{\mathbf{x}}_{ir}^{\tau-}(k) + \mathbf{H}_{ir}(k)^T \mathbf{B}_{ir} \mathbf{z}_{ir}(k) \\ &= \frac{1}{N} \mathbf{W}_{ir}^-(k) \hat{\mathbf{x}}_{ir}^{\tau-}(k) + b \begin{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) \\ \mathbf{0} \end{bmatrix} \mathbf{z}_{ir}(k) \end{aligned} \quad (2.14)$$

We note that in (2.14) the quantity $\mathbf{z}_{ir}(k)$ is not actually measured but, using (2.12), one obtains

$$\begin{aligned} \mathbf{v}_{ir}^0 &= \frac{1}{N} \mathbf{W}_{ir}^-(k) \hat{\mathbf{x}}_{ir}^{\tau-}(k) + b \begin{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) \\ \mathbf{0} \end{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) \mathbf{p}_i(k) = \\ &= \frac{1}{N} \mathbf{W}_{ir}^-(k) \hat{\mathbf{x}}_{ir}^{\tau-}(k) + b \begin{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) \\ \mathbf{0} \end{bmatrix} \mathbf{p}_i(k) \in \mathbb{R}^6 \end{aligned} \quad (2.15)$$

which only depends on known quantities.

The update passage for \mathbf{V}_{ir}^0 applied to our case reads as:

$$\begin{aligned} \mathbf{V}_{ir}^0 &= \frac{1}{N} \mathbf{W}_{ir}^-(k) + \mathbf{H}_{ir}(k)^T \mathbf{B}_{ir} \mathbf{H}_{ir}(k) = \\ &= \frac{1}{N} \mathbf{W}_{ir}^-(k) + b \begin{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{6 \times 6} \end{aligned} \quad (2.16)$$

Algorithm 1 - ICF at drone i relative to target r at time step k

Input: Prior state estimate $\hat{\mathbf{x}}_{ir}^{\tau-}(k)$, prior information matrix $\mathbf{W}_{ir}^-(k)$, observation matrix \mathbf{H}_{ir} , consensus rate parameter ϵ , total consensus iterations K and process covariance \mathbf{Q} .

1) Get measurement vector \mathbf{z}_{ir} and matrix \mathbf{B}_{ir}

2) Compute initial information matrix \mathbf{V}_{ir}^0 and vector \mathbf{v}_{ir}^0

$$\mathbf{V}_{ir}^0 \leftarrow \frac{1}{N} \mathbf{W}_{ir}^-(k) + b\mu_{ir}(k) \begin{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (2.4)$$

$$\mathbf{v}_{ir}^0 \leftarrow \frac{1}{N} \mathbf{W}_{ir}^-(k) \hat{\mathbf{x}}_{ir}^{\tau-}(k) + b\mu_{ir}(k) \begin{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) \\ \mathbf{0} \end{bmatrix} \mathbf{p}_i(k) \quad (2.5)$$

3) Perform average consensus on \mathbf{V}_{ir}^0 and \mathbf{v}_{ir}^0 independently

for $\kappa = 1$ **to** K **do**

a) Send $\mathbf{V}_{ir}^{\kappa-1}$ and $\mathbf{v}_{ir}^{\kappa-1}$ to all neighbors $j \in \mathcal{N}_i$

b) Receive $\mathbf{V}_{ir}^{\kappa-1}$ and $\mathbf{v}_{ir}^{\kappa-1}$ from all neighbors $j \in \mathcal{N}_i$

c) Update:

$$\mathbf{V}_{ir}^{\kappa} \leftarrow \mathbf{V}_{ir}^{\kappa-1} + \epsilon \sum_{j \in \mathcal{N}_i} (\mathbf{V}_j^{\kappa-1} - \mathbf{V}_{ir}^{\kappa-1}) \quad (2.6)$$

$$\mathbf{v}_{ir}^{\kappa} \leftarrow \mathbf{v}_{ir}^{\kappa-1} + \epsilon \sum_{j \in \mathcal{N}_i} (\mathbf{v}_j^{\kappa-1} - \mathbf{v}_{ir}^{\kappa-1}) \quad (2.7)$$

end

4) Compute a posteriori state estimate and information matrix for time k

$$\hat{\mathbf{x}}_{ir}^{\tau+}(k) \leftarrow (\mathbf{V}_{ir}^K)^\dagger \mathbf{v}_{ir}^K \quad (2.8)$$

$$\mathbf{W}_{ir}^+ \leftarrow N \mathbf{V}_{ir}^K \quad (2.9)$$

5) Predict for next time step ($k+1$)

$$\mathbf{W}_{ir}^-(k+1) \leftarrow ((\mathbf{A}_d^{-T} \mathbf{W}_{ir}^+(k) \mathbf{A}_d^{-1})^\dagger + \mathbf{Q})^{-1} \quad (2.10)$$

$$\hat{\mathbf{x}}_{ir}^{\tau-}(k+1) \leftarrow \mathbf{A}_d \hat{\mathbf{x}}_{ir}^{\tau+}(k) \quad (2.11)$$

Output: State estimate $\hat{\mathbf{x}}_{ir}^{\tau+}(k)$ and information matrix $\mathbf{W}_{ir}^+(k)$.

In (2.4) and (2.5), $\mu_{ir}(k) = 1$ if a measurement is available at instant k or $\mu_{ir}(k) = 0$ otherwise.

Algorithm (1) differs from the original algorithm presented in [4] also for steps 4 and 5. Indeed, equations (2.8) and (2.10) have been slightly modified because, due to the particular structure of \mathbf{H}_{ir} , the full information

matrix can be singular at system startup, so we cannot use its inverse. This singularity is then avoided as with step 5 the knowledge about the model is used and the information matrix is updated also in the block relative to the velocity of the target (which is not measured directly from the camera). Notice that for the pseudoinverse of a product, in general, it does not hold the usual formula similar to inverse of product of matrices.

The outputs of the ICF at each iteration are the estimate of the state of the r -th target

$$\hat{\mathbf{x}}_{ir}^{\tau+}(k) = \begin{bmatrix} \hat{\mathbf{p}}_{ir}^{\tau+}(k) \\ \hat{\mathbf{v}}_{ir}^{\tau+}(k) \end{bmatrix}$$

composed by the estimated position and velocity, respectively, and the corresponding information matrix \mathbf{W}_{ir}^+ . Recall that the target model assumed by the drones is a constant velocity model, hence, if a target moves with acceleration different from zero, we expect a non negligible estimation error. Moreover, as mentioned before, the algorithm is extended to handle the multitarget case by simply running the same steps of Algorithm 1 for each target, hence obtaining M state estimations. Another possible solution could be to run a single instance of the ICF with an enlarged state which aggregates all the states of the targets. In the first case we would have different variables for each target, while in the latter case we will manage stack vectors and block matrices. In both cases, the mathematical result will be the same, hence the chosen solution will not change the final result and is only a matter of implementation.

2.2 ICF for the Alternative Target Model

If the single integrator model is used for the targets (1.5), the discrete-time model considered for the filter is

$$\mathbf{p}_r^\tau(k+1) = \mathbf{p}_r^\tau(k) + \Delta T \mathbf{u}_r^\tau(k) + \gamma(k) \quad r = 1, \dots, M \quad (2.17)$$

where ΔT is the discretization step, while $\gamma(k) \in \mathcal{N}(\mathbf{0}, \mathbf{Q})$ is gaussian process noise with positive definite covariance matrix \mathbf{Q} .

In this case, the observation matrix is simply

$$\mathbf{H}_{ir} = \mathbf{\Pi}_{\beta_{ir}}(k) \in \mathbb{R}^{3 \times 3} \quad (2.18)$$

and thus the update passages for \mathbf{v}_{ir}^0 and \mathbf{V}_{ir}^0 are modified as

$$\mathbf{v}_{ir}^0 = \frac{1}{N} \mathbf{W}_{ir}^-(k) \hat{\mathbf{p}}_{ir}^{\tau-}(k) + b \mathbf{\Pi}_{\beta_{ir}}(k) \mathbf{p}_i(k) \in \mathbb{R}^3 \quad (2.19)$$

and

$$\mathbf{V}_{ir}^0 = \frac{1}{N} \mathbf{W}_{ir}^-(k) + b \mathbf{\Pi}_{\beta_{ir}}(k) \in \mathbb{R}^{3 \times 3} \quad (2.20)$$

and the filter that drone i runs for the j -th target works with the state $\hat{\mathbf{p}}_{ir}(k) \in \mathbb{R}^3$. In the second step of the filter, the projection matrix is multiplied by $\mu_{ir}(k)$, defined previously.

Moreover, step 5 of Algorithm 1 is simplified as

$$\mathbf{W}_{ir}^-(k+1) \leftarrow \left(\left(\mathbf{W}_{ir}^+(k) \right)^\dagger + \mathbf{Q} \right)^{-1} \quad (2.21)$$

$$\hat{\mathbf{p}}_{ir}^{\tau-}(k+1) \leftarrow \hat{\mathbf{p}}_{ir}^{\tau+}(k) + \Delta T \mathbf{u}(k) \quad (2.22)$$

In the following, the double integrator model is preferred because it requires less communication between drones and targets, since the velocity input of the targets is not sent, but estimated.

2.3 ICF Stability

The stability of the filter is proved for the case in which the model of the targets is the double integrator. However, the proof can be easily modified for the single integrator case, considering the different definition of \mathbf{H}_{ir} .

To show the stability of the employed filter we make the following assumptions:

Assumption 1. *No collision drone-target occurs, so that the bearing measurements are always well-defined.*

Assumption 2. *The target state is collectively observable, i.e. the discrete-time Observability Gramian is full-rank*

$$\frac{1}{K} \sum_{k=0}^K \sum_{i=1}^N \left(\mathbf{A}_d^k \right)^T \begin{bmatrix} \mathbf{\Pi}_{\beta_{ir}}(k) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{A}_d^k > \mu_1 \mathbf{I}$$

for some $\mu_1 > 0$, which reduces to $\frac{1}{K} \sum_{k=0}^K \sum_{i=1}^N \mathbf{\Pi}_{\beta_{ir}}(k) > \mu_2 \mathbf{I}$ for some $\mu_2 > 0$.

Assumption 3. *The information matrix is initialized such that $\mathbf{W}_{ir}(0) > \mu_3 \mathbf{I}$ for some $\mu_3 > 0$, $i = 1, \dots, N$.*

Remark 1. *Assumption 2 is satisfied either if there is a single persistently exciting direction β_{ir} or at least two non-collinear directions β_{ir} and β_{jr} , as explained in [31].*

Assumption 1 can be trivially met by adding a constraint on the UAV-target minimum distance (as done in the case study of this work), Assumption 2 is a Persistency of Excitation (PE) condition that will be actually enforced at runtime by the algorithm proposed in the following Chapter, Assumption 3 is only an initialization.

Under these assumptions, the proof provided in [3] holds, stating that the weighted squared error vector converges to zero exponentially fast.

In more detail, to prove the stability of the filter we start by defining:

$$L_{ir}(k) = \mathbf{e}_{ir}(k)^T \mathbf{W}_{ir}^-(k) \mathbf{e}_{ir}(k), \quad (2.23)$$

where $\mathbf{e}_{ir}(k) = \mathbf{x}_r^\tau(k) - \hat{\mathbf{x}}_{ir}^{\tau-}$. Consider the stack:

$$\mathbf{L}_r(k) = \begin{bmatrix} L_{1r}(k) & \dots & L_{Nr}(k) \end{bmatrix}^T \quad (2.24)$$

Under the previous Assumptions, we can use the proof of Theorem 5 in [3], which shows that:

$$\mathbf{L}_r(k+1) \leq \beta \mathbf{P} \mathbf{L}_r(k) \quad (2.25)$$

where $\beta < 1$ and \mathbf{P} is a primitive matrix [35] used for the consensus step. We used $\mathbf{P} = \mathbf{I} - \epsilon \mathbf{L}$, with \mathbf{L} being the laplacian matrix of the unweighted graph [35], which is primitive for $0 < \epsilon < \frac{1}{\max_{i \in \mathcal{V}} \Delta_i}$, where Δ_i is the degree of vertex i . Being \mathbf{P} primitive and $\beta < 1$, it follows that the matrix $\beta \mathbf{P}$ is Schur stable, meaning that the components of \mathbf{L}_r goes to zero exponentially fast. Then, as, due to Assumptions 2 and 3, the information matrices \mathbf{W}_{ir} are bounded from below, it follows that estimation errors goes to zero exponentially fast and the observer is uniformly globally exponentially stable. Consider that, in general, the real velocity of the target will not be constant, but the observer is input-to-state stable with respect to perturbations in the velocity dynamics, hence bounded accelerations will cause bounded estimation errors.

As mentioned before, Assumption 2 is a Persistency of Excitation (PE) condition. In Chapter 3 we present an algorithm which guarantees that a prescribed level of PE is always maintained, satisfying in this way Assumption 2.

Chapter 3

Target Monitoring

In this chapter, the main tools that will be used for the main results are introduced first, then these concepts are combined in order to guarantee that the PE condition is satisfied, thus solving the persistent target monitoring problem. The tools that will be introduced are: the weighted Observability Gramian (OG) with forgetting factor, the perception awareness weights and the High Order Control Barrier Functions (HOCBFs).

3.1 Information Measure

The Observability Gramian (OG) is a tool used to study the observability of linear time-varying systems (e.g., a nonlinear system linearized along the nominal trajectory) as well as the local observability of nonlinear systems. The observability property is linked to the observation (reconstruction) of the state of a system from its output. The system is locally observable if the OG is full rank. Its expression is

$$\mathbf{G}(t_0, t_f) \triangleq \int_{t_0}^{t_f} \Phi(\tau, t_0)^T \mathbf{H}(\tau)^T \mathbf{H}(\tau) \Phi(\tau, t_0) d\tau \quad (3.1)$$

where $\mathbf{H}(\cdot)$ is the observation matrix of the system, $\Phi(\cdot)$ is the state transition matrix of the system, namely the matrix satisfying

$$\dot{\Phi}(t, t_0) = \mathbf{A}(t)\Phi(t, t_0), \quad \Phi(t_0, t_0) = \mathbf{I} \quad (3.2)$$

being $\mathbf{A}(\cdot)$ the Jacobian $\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$.

The OG also finds application in the field of *active sensing*, where it is used in order to quantify the acquired information [11], [36], [37]. Usually, in active sensing, the task is formulated as an optimization problem, thus a scalar

cost function of the OG (or, more generally, of an information matrix) is required. Common metrics for information maximisation (error covariance minimization) are the following [38], [39]:

- **A-criterion:** it corresponds to the trace of the inverse of an information matrix or the trace of a covariance. This is linked to the minimization of the average uncertainty
- **E-criterion:** it corresponds to the minimum eigenvalue of an information matrix or the maximum eigenvalue of a covariance matrix. This is linked to the minimization of the length of the largest axis of the uncertainty ellipsoid
- **D-criterion:** it corresponds to the determinant of an information matrix or of a covariance matrix. This is linked to the minimization of the volume of the uncertainty ellipsoid.

Notice that, however, in the problem considered in this Thesis the goal is not the maximization of the information (minimization of the error), but the maintenance of the information above a positive threshold. Thus, the scalar metrics listed above will be used in a different way with respect to usual active sensing problem.

In the case under exam, the state transition matrix, as will be clear later, is simply the identity matrix and, considering the observation matrix defined in (2.13), the OG representing the information acquired about the target position until time t , indicated as $\mathbf{G}_r(t_0, t) \in \mathbb{R}^{3 \times 3}$, can be expressed as:

$$\mathbf{G}_r = \int_{t_0}^t \sum_{i=1}^N \mathbf{\Pi}_{\beta_{ir}}^T \mathbf{\Pi}_{\beta_{ir}} d\tau = \int_{t_0}^t \sum_{i=1}^N \mathbf{\Pi}_{\beta_{ir}} d\tau \quad (3.3)$$

where the orthogonal projector properties were used to simplify the expression. The OG is a positive semi-definite matrix and it is invertible if and only if the position of the target is observable along the trajectory. In reason of this, the minimum eigenvalue λ_{1r} (E-criterion) of this matrix can be taken as observability metric: it quantifies how far is the position of the target from being unobservable.

As the integrand of the OG is a positive semi-definite matrix, the OG is monotonically increasing in time, for this reason, in order to achieve the desired goal of always maintaining a certain level of PE, we introduce a *forgetting factor* in its dynamics, which makes the information decrease if there are no new measurement. Also, in order to take into account the sensing limits of the drones, we introduce weights in the information acquired at time t . The dynamics of the weighted OG with forgetting factor [40] can

then be written as:

$$\dot{\mathbf{G}}_r = -\rho \mathbf{G}_r + b \sum_{i=1}^N w_{ir} \mathbf{\Pi}_{\beta_{ir}} \quad (3.4)$$

where $\rho > 0$ is the forgetting factor, $w_{ir} \in [0, 1]$ is a weight whose expression will be defined later and which is used to encode sensing constraints in the information dynamics and, as before, b is the information gain associated to the measurement. Notice that (3.4) has a closed form solution [40]:

$$\mathbf{G}_r(t) = e^{-\rho(t-t_0)} \mathbf{G}_r(t_0) + b \int_{t_0}^t e^{-\rho(t-\tau)} \sum_{i=1}^N w_{ir} \mathbf{\Pi}_{\beta_{ir}}(\tau) d\tau \quad (3.5)$$

In this way, if the target starting from a certain point is no more observed, then the information will decay to zero exponentially fast.

3.2 Perception Awareness

As mentioned in the previous sections, the drones are made aware about the sensing limitations by weighting the OG. In (3.4), indeed, w_{ir} is a scalar differentiable quantity used by the i -th drone to weight the information acquired about the r -th target and thus to make the drone aware of its perception limits. For this reason, it is also called *perception awareness weight*.

The weight w_{ir} smoothly varies from 1 inside the sensing limit region to 0 outside the sensing limit region. The information artificially decreases in case the target approaches the maximum sensing range or angle of the FoV. The weight is defined as

$$w_{ir} = w_{Dir} w_{\beta_{ir}} \quad (3.6)$$

with

$$w_{Dir} = \begin{cases} e^{-\frac{(\hat{d}_{ir} - D_m^{th})^2}{\sigma_{D_m}^2}}, & \text{if } \hat{d}_{ir} < D_m^{th} \\ 1, & \text{if } D_m^{th} \leq \hat{d}_{ir} \leq D_M^{th} \\ e^{-\frac{(\hat{d}_{ir} - D_M^{th})^2}{\sigma_{D_M}^2}}, & \text{if } \hat{d}_{ir} > D_M^{th} \end{cases} \quad (3.7)$$

$$w_{\beta_{ir}} = \begin{cases} e^{-\frac{(c_{ir} - \cos(\alpha_M^{th}))^2}{\sigma_{\beta}^2}}, & \text{if } c_{ir} < \cos(\alpha_M^{th}) \\ 1, & \text{if } c_{ir} \geq \cos(\alpha_M^{th}) \end{cases}$$

where $\hat{d}_{ir} = \|\hat{\mathbf{p}}_{ir}^{\tau+} - \mathbf{p}_i\|$ is the *estimated* distance from the i -th robot to the position of the r -th target, $D_m^{th}, D_M^{th}, \alpha_M^{th}$ are parameters that represent respectively the distance and FoV angle at which the weight start to decrease,

$\sigma_{D_M}, \sigma_{D_m}, \sigma_\beta$ are standard deviations, $c_{ir} = -\beta_{ir} \mathbf{e}_3$ is the cosine of the angle between the bearing and the negative z -axis of the frame attached to the i -th drone. As it can be noticed from Fig. 3.1, the weights $w_{D_{ir}}$ and $w_{\beta_{ir}}$ do not vanish to zero as the sensing limits are approaching. The idea behind this design choice is to allow the robots losing a measurement when enough information is available, for then going back later exploiting the non-zero gradient of the weights. This choice also allows quadrotors which are not currently acquiring measurements to use the group-level knowledge about the target position, and their own weight gradient, to obtain a measurement in the future. This feature is possible because the drones compute the estimated positions of the targets, which can be used to compute the distances and the bearings also when the latter are not directly obtained from the camera.

How long it takes for these weights to decrease to zero is a design choice which also depends on the application. The perception awareness weights are indeed very important for the final result and they have to be tuned carefully.

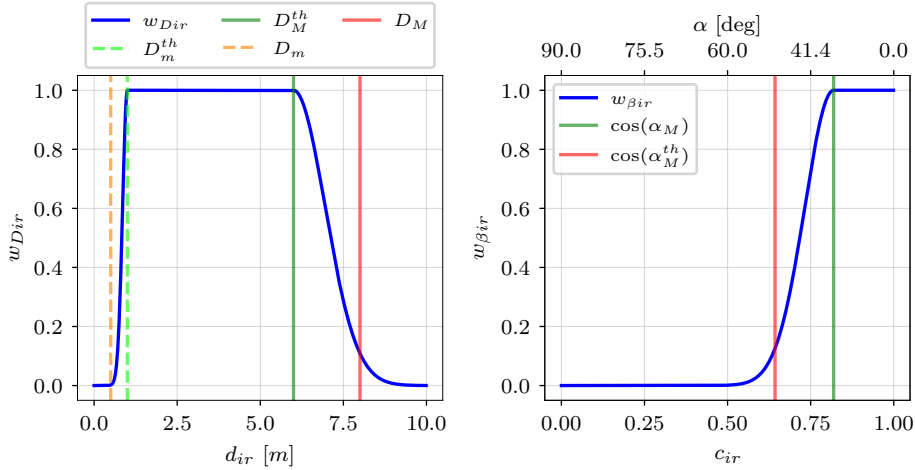


Figure 3.1: Perception awareness weights for values $D_m = 0.5$ m, $D_m^{th} = 1$ m, $D_M = 8$ m, $D_M^{th} = 6$ m, $\sigma_{D_M}^2 = 1.8m^2$, $\sigma_{D_m}^2 = 0.04m^2$, $\alpha_M = 50$ deg, $\alpha_M^{th} = 35$ deg, $\sigma_\beta^2 = 0.015$ rad².

3.3 High Order Control Barrier Functions

Control Barrier Functions (CBFs) are a tool to enforce safety constraints in dynamical systems. Only the basic concepts are reported here. For a complete explanation about CBFs the reader is referred to [41] and [21], while some examples of applications to multi-agent systems can be found in

[28] for collision avoidance and in [29] for connectivity maintenance.

Consider the (nonlinear) system in control affine form

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})\mathbf{u} \quad (3.8)$$

with state $\mathbf{x} \in \mathbb{R}^n$ and input $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^m$, f and g locally Lipschitz. The basic idea of CBFs is to enforce the forward invariance of a safe set \mathcal{C} , defined as the superlevel set of a continuously differentiable scalar function $h(\mathbf{x}) : \mathcal{D} \subset \mathbb{R}^n \rightarrow \mathbb{R}$

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{D} \subset \mathbb{R}^n : h(\mathbf{x}) \geq 0\} \quad (3.9)$$

The function h is a Control Barrier Function if there exists an extended class \mathcal{K} function α such that for the system (3.8):

$$\sup_{\mathbf{u} \in \mathcal{U}} [L_f h(\mathbf{x}) + L_g h(\mathbf{x})\mathbf{u} + \alpha(h(\mathbf{x}))] \geq 0 \quad (3.10)$$

for all $\mathbf{x} \in \mathcal{D}$. In (3.10) L_f and L_g represents Lie (directional) derivatives of $h(\mathbf{x})$ along the vector fields \mathbf{f} and \mathbf{g} respectively for the system (3.8), while an extended class \mathcal{K} function $\alpha : \mathbb{R} \rightarrow \mathbb{R}$ is strictly increasing and $\alpha(0) = 0$. As a consequence, $\alpha(h(\cdot))$ preserves the sign of h . Note that in this work, with the term CBFs, we mean *zeroing* CBFs, as defined in [41]. It has been shown [23], [27] that any locally Lipschitz continuous controller $\mathbf{u}(\mathbf{x})$ satisfying the CBF constraint (3.10) renders the set \mathcal{C} forward invariant and, if \mathcal{C} is compact, asymptotically stable. The simplest and most common choice for the function $\alpha(\cdot)$ is to take a linear function $\alpha(h(\mathbf{x})) := \alpha h(\mathbf{x})$, with a slight abuse of notation.

To design a safe controller given the system (3.8) with a *potentially* unsafe input \mathbf{u}_d , namely an input which could drive the system outside the safe set, we consider the Quadratic Program (QP) with the safe condition (3.10) as linear inequality constraint

$$\begin{aligned} \min_{\mathbf{u} \in \mathcal{U}} \quad & \frac{1}{2} \|\mathbf{u} - \mathbf{u}_d\|_2^2 \\ \text{s.t.} \quad & L_f h(\mathbf{x}) + L_g h(\mathbf{x})\mathbf{u} \geq -\alpha(h(\mathbf{x})) \end{aligned} \quad (3.11)$$

which provides the smallest modification to \mathbf{u}_d for coping with the CBF constraint and is thus a minimally invasive controller. It is also possible to unify safety and stability within the same QP using Control Lyapunov Functions (CLFs) [21], [41]. In this case, the safety constraints (enforced by CBFs) are hard constraints, while the stability constraints (enforced by CLFs) are made soft by relaxation variables, indeed one can easily understand that, in most cases, safety is more important than performance. In this work, the

CLFs will not be exploited, but the concepts of relaxation variables and soft constraints will be used to achieve the main results.

The CBFs discussed so far can be used to enforce safety constraints in systems of relative degree 1, namely systems in which the first derivative of the CBF depends explicitly on the input. In many systems (such as the one addressed in this work), however, this assumption does not hold and the concept of CBF must be thus extended to systems with arbitrary high relative degree. Possible solutions to this problem are Exponential Control Barrier Functions (ECBFs) [24] and High Order Control Barrier Functions (HOCBFs) [25] [26] [27]. The latter are more general than ECBFs, and thus preferable. Following the HOCBFs approach, given a r -th order differentiable function $h : \mathbb{R}^n \rightarrow \mathbb{R}$ and r extended class \mathcal{K} functions $\alpha_1, \dots, \alpha_r$, we define a series of functions with the corresponding superlevel sets as

$$\begin{aligned} \psi_0(\mathbf{x}) &= h(\mathbf{x}), & \mathcal{C}_0 &= \{\mathbf{x} \in \mathbb{R}^n : \psi_0(\mathbf{x}) \geq 0\} \\ \psi_1(\mathbf{x}) &= \dot{\psi}_0(\mathbf{x}) + \alpha_1(\psi_0(\mathbf{x})), & \mathcal{C}_1 &= \{\mathbf{x} \in \mathbb{R}^n : \psi_1(\mathbf{x}) \geq 0\} \\ & \vdots & & \\ \psi_r(\mathbf{x}) &= \dot{\psi}_{r-1}(\mathbf{x}) + \alpha_r(\psi_{r-1}(\mathbf{x})), & \mathcal{C}_r &= \{\mathbf{x} \in \mathbb{R}^n : \psi_r(\mathbf{x}) \geq 0\} \end{aligned} \quad (3.12)$$

Given the functions and sets defined in (3.12), h is a High Order (zeroing) Control Barrier Function of order r for the dynamical system (3.8) if there exist r extended class \mathcal{K} functions $\alpha_1, \dots, \alpha_r$ and an open set \mathcal{D} such that

$$\sup_{\mathbf{u} \in \mathcal{U}} \psi_r(\mathbf{x}) = \sup_{\mathbf{u} \in \mathcal{U}} [L_f \psi_{r-1}(\mathbf{x}) + L_g \psi_{r-1}(\mathbf{x}) \mathbf{u} + \alpha_r(\psi_{r-1}(\mathbf{x}))] \geq 0 \quad (3.13)$$

for all $\mathbf{x} \in \mathcal{C} = \mathcal{C}_0 \cap \mathcal{C}_1 \cap \dots \cap \mathcal{C}_{r-1} \subset \mathcal{D} \subset \mathbb{R}^n$. Note that here we don't use the concept of *least* relative degree introduced in [27] and we consider the relative degree of h to be uniform. Note also that, as explained in [27], thanks to the use of the extended class \mathcal{K} functions α_k , this formulation is more robust than [25] because the functions are well defined also outside the safe set and hence the system can handle perturbations due to noise or model uncertainties leading to $\psi_{k-1}(\mathbf{x}) < 0$, $1 \leq k \leq r$ and could start from outside the safe set $\mathcal{D} \setminus \mathcal{C}$.

To use HOCBFs to enforce safety in systems with high relative degree $r > 1$, the QP (3.11) is modified as

$$\begin{aligned} \min_{\mathbf{u} \in \mathcal{U}} \quad & \frac{1}{2} \|\mathbf{u} - \mathbf{u}_d\|_2^2 \\ \text{s.t.} \quad & L_f \psi_{r-1}(\mathbf{x}) + L_g \psi_{r-1}(\mathbf{x}) \mathbf{u} + \alpha_r(\psi_{r-1}(\mathbf{x})) \geq 0 \end{aligned} \quad (3.14)$$

where the HOCBF condition is again a linear inequality constraint.

3.4 Persistent Target Monitoring

In this section, the concepts introduced earlier are combined and applied to the persistent monitoring problem in order to guarantee that the PE conditions are satisfied. This is achieved applying the HOCBFs to the distributed system and using the minimum eigenvalue of the OG with forgetting factor as information measure (E-criterion).

The centralized problem is presented first, and it is later decentralized and solved in a distributed way.

3.4.1 Centralized Problem

The goal is to make sure that Assumption 2 remains valid. This can be seen as a safety constraint and it can be satisfied by ensuring that the minimum eigenvalue λ_{1r} of the OG remains over a certain positive threshold. For this purpose, the safe set is defined as:

$$\mathcal{C}_r = \{\mathbf{x} \in \mathbb{R}^{3N+(3+9)} : h_r(\mathbf{x}) = \lambda_{1r}(\mathbf{x}) - \epsilon \geq 0\} \quad (3.15)$$

It is defined here the set corresponding to one target, adding another target is done by considering the intersection of this set with the set corresponding to the other target, as shown later. In (3.15), the minimum eigenvalue of \mathbf{G}_r is indicated as λ_{1r} , ϵ is a positive threshold, while the state \mathbf{x} is represented by the stack of the quadrotors position, the estimated target position and the vectorized OG, i.e.

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{p}_r^\top \\ \text{vec}(\mathbf{G}_r) \end{bmatrix} \in \mathbb{R}^{3N+(3+9)} \quad (3.16)$$

where $\text{vec}(\cdot)$ is the vectorization operator and \mathbf{p} aggregate all the quadrotors positions.

The corresponding system dynamics is given by:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t) + \mathbf{g}(\mathbf{x})\mathbf{u} \quad (3.17)$$

where:

$$\mathbf{f}(\mathbf{x}, t) = \begin{bmatrix} \mathbf{0} \\ \mathbf{v}_r^\top(t) \\ \text{vec}\left(-\rho\mathbf{G}_r + \sum_{i=1}^N bw_{ir}\mathbf{\Pi}_{\beta_{ir}}\right) \end{bmatrix} \in \mathbb{R}^{3N+(3+9)} \quad (3.18)$$

where, with an abuse of notation, only the direct time dependency is indicated, and

$$\mathbf{g}(\mathbf{x}) = \begin{bmatrix} (\mathbf{1}_N \otimes \mathbf{I}_3) \\ \mathbf{0}_{3 \times 3N} \\ \mathbf{0}_{9 \times 3N} \end{bmatrix} \in \mathbb{R}^{(3N+(3+9)) \times 3N} \quad (3.19)$$

with \otimes representing the Kronecker product.

One can verify that the function $h_r(\mathbf{x})$ has relative degree 2 w.r.t. the system inputs \mathbf{u} , because $L_g(h_r(\mathbf{x})) = 0$, hence the need to resort to the HOCBFs. In our case

$$\begin{aligned}\psi_{0r}(\mathbf{x}) &= h_r(\mathbf{x}) = \lambda_{1r}(\mathbf{x}) - \epsilon \\ \psi_{1r}(\mathbf{x}) &= L_f(h_r(\mathbf{x})) = \frac{\partial \lambda_{1r}(\mathbf{x})}{\partial(\text{vec}(\mathbf{G}_r))} \text{vec}(\dot{\mathbf{G}}_r) + (\lambda_{1r} - \epsilon)\end{aligned}\quad (3.20)$$

where as extended class \mathcal{K} function $\alpha_1(\cdot) = \text{Id}(\cdot)$ (the identity function) is used, and $\frac{\partial \lambda_{1r}}{\partial(\text{vec}(\mathbf{G}_r))} = \mathbf{v}_{1r}^T \otimes \mathbf{v}_{1r}^T$, with \mathbf{v}_{1r} being the eigenvector of the OG associated to λ_{1r} (Theorem 1 of [42]). The complete computations can be found in Appendix B.

The centralized QP which needs to be solved is the following:

$$\begin{aligned}\min_{\mathbf{u} \in \mathcal{U}} \quad & \|\mathbf{u} - \mathbf{u}_d\|_2^2 \\ \text{s.t.} \quad & \sum_{i=1}^N L_{gi} \psi_{1r}(\mathbf{x}) \mathbf{u}_i + L_f \psi_{1r}(\mathbf{x}) + \psi_{1r}(\mathbf{x}) \geq 0\end{aligned}\quad (3.21)$$

Where, again, as extended class \mathcal{K} function simply the identity is chosen. Also,

$$L_{gi} \psi_{1r}(\mathbf{x}) = b \mathbf{v}_{1r}^T \otimes \mathbf{v}_{1r}^T \left(\text{vec}(\mathbf{\Pi}_{\beta_{ir}}) \frac{\partial w_{ir}}{\partial \mathbf{p}_i} + w_{ir} \frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial \mathbf{p}_i} \right) \quad (3.22)$$

and

$$\begin{aligned}L_f \psi_{1r}(\mathbf{x}) &= \left(\text{vec}(\dot{\mathbf{G}}_r)^T \frac{\partial^2 \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)^2} + (\alpha - \rho) \mathbf{v}_{1r}^T \otimes \mathbf{v}_{1r}^T \right) \text{vec}(\dot{\mathbf{G}}_r) + \\ &+ b \mathbf{v}_{1r}^T \otimes \mathbf{v}_{1r}^T \sum_{i=1}^N \left(\text{vec}(\mathbf{\Pi}_{\beta_{ir}}) \frac{\partial w_{ir}}{\partial \mathbf{p}_i^T} + w_{ir} \frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial \mathbf{p}_i^T} \right) \mathbf{v}_r^T = \\ &= c(\mathbf{x}) + \sum_{i=1}^N d_i(\mathbf{x})\end{aligned}\quad (3.23)$$

where $L_f \psi_{1r}(\mathbf{x})$ is split in the separable part $\sum_{i=1}^N d_i(\mathbf{x})$ and the non-separable one $c(\mathbf{x})$ (the reason will be clear later). Again, the full computations with explanation are in Appendix B. In (3.23), $c(\mathbf{x})$ and $d_i(\mathbf{x})$ have the following expressions

$$c(\mathbf{x}) := \left(\text{vec}(\dot{\mathbf{G}}_r)^T \frac{\partial^2 \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)^2} + (1 - \rho) \mathbf{v}_{1r}^T \otimes \mathbf{v}_{1r}^T \right) \text{vec}(\dot{\mathbf{G}}_r) \quad (3.24)$$

and

$$d_i(\mathbf{x}) := b\mathbf{v}_{1r}^T \otimes \mathbf{v}_{1r}^T \left(\text{vec}(\mathbf{\Pi}_{\beta_{ir}}) \frac{\partial w_{ir}}{\partial \mathbf{p}_r^\tau} + w_{ir} \frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial \mathbf{p}_r^\tau} \right) \mathbf{v}_r^\tau \quad (3.25)$$

where $\frac{\partial^2 \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)^2}$ can be computed as follows:

$$\frac{\partial^2 \lambda_1}{\partial \text{vec}(\mathbf{G}_r)^2} = \mathbf{K}_3 \left(\mathbf{Y}_{1r}^\dagger \otimes \mathbf{v}_{1r} \mathbf{v}_{1r}^T + \mathbf{v}_{1r} \mathbf{v}_{1r}^T \otimes \mathbf{Y}_{1r}^\dagger \right) \quad (3.26)$$

with \dagger indicating the Moore-Penrose pseudo-inverse, $\mathbf{Y}_{1r} := \lambda_{1r} \mathbf{I} - \mathbf{G}_r$ and \mathbf{K}_3 being the commutation matrix (see Theorem 4 of [42] and Appendix B).

The QP problem in (3.21) is centralized. The objective is for each drone to solve a local QP, using only local quantities, so that the collective solution of the local QPs results in the satisfaction of the centralized constraint in (3.21). In the next section, the distributed solution is shown.

3.4.2 Distributed Problem

In the previous section, the constraint which needs to be enforced in order to ensure satisfaction of Assumption 2 have been introduced. It is now shown how to guarantee its satisfaction in a distributed way. First, notice from (3.23) that $L_f \psi_{1r}(\mathbf{x})$ can be split in a part which is local to each robot, $d_i(\mathbf{x})$, and a part which is not already separate $c(\mathbf{x})$. A possible strategy to satisfy the previous criterion is for each drone to solve the following QP:

$$\begin{aligned} \min_{\mathbf{u}_i \in \mathcal{U}_i} \quad & \|\mathbf{u}_i - \mathbf{u}_{id}\|_2^2 \\ \text{s.t.} \quad & L_{g_i} \psi_{1r}(\mathbf{x}) \mathbf{u}_i + d_i(\mathbf{x}) \geq -k_i(\mathbf{x}) (c(\mathbf{x}) + \psi_{1r}(\mathbf{x})) \end{aligned} \quad (3.27)$$

where the weights $k_i(\mathbf{x})$ (not necessarily state dependent) satisfy the following conditions (i.e. they form a convex combination):

- $k_i(\mathbf{x}) \geq 0$
- $\sum_{i=1}^N k_i(\mathbf{x}) = 1$

Then, using the previous inequalities in the original constraints taking the summation for each robot of the left hand side of the inequality in (3.27), one obtains

$$\begin{aligned} \sum_{i=1}^N L_{g_i} \psi_{1r}(\mathbf{x}) \mathbf{u}_i + \sum_{i=1}^N d_i(\mathbf{x}) &\geq - \left(\sum_{i=1}^N k_i(\mathbf{x}) \right) (c(\mathbf{x}) + \psi_{1r}(\mathbf{x})) = \\ &= - (c(\mathbf{x}) + \psi_{1r}(\mathbf{x})) \end{aligned} \quad (3.28)$$

which satisfies the centralized constraint in (3.21).
The most trivial choice for the weights is

$$k_i(\mathbf{x}) = \frac{1}{N}$$

which divides equally the constraint among the robots, but other choices are possible [30].

In the local constraint in (3.27), some of the variables are not immediately locally available, but they can be estimated in a decentralized way:

- $\sum_{i=1}^N w_{ir} \mathbf{\Pi}_{\beta_{ir}}$: this quantity appears in $\text{vec}(\dot{\mathbf{G}}_r)$ and can be computed in a distributed way by dynamic average consensus [13] [14] (3.30) and multiplying the average by the number of quadrotors N , which is known.
- \mathbf{G}_r : Each drone has its own copy \mathbf{G}_{ir} of the information collected about the target \mathbf{G}_r . Starting from the same initial conditions, each drone has the same dynamics for \mathbf{G}_{ir} up to the consensus error on $\sum_{i=1}^N w_{ir} \mathbf{\Pi}_{\beta_{ir}}$. In order to have consistency across the network we add a consensus term to the OG dynamics:

$$\dot{\mathbf{G}}_{ir} = -\rho \mathbf{G}_{ir} + \sum_{i=1}^N b w_{ir} \mathbf{\Pi}_{\beta_{ir}} + \sum_{j \in \mathcal{N}_i} (\mathbf{G}_{jr} - \mathbf{G}_{ir}). \quad (3.29)$$

Then, also the quantities λ_{1r} and \mathbf{v}_{1r} are known to all the quadrotors.

- $\mathbf{p}_r^\tau, \mathbf{v}_r^\tau$: every drone has its own estimate $\hat{\mathbf{p}}_{ir}^{\tau+}$ and $\hat{\mathbf{v}}_{ir}^{\tau+}$ given by the ICF. These quantities converge to the same value due to the consensus iterations in the filter.

If the single integrator is used to model the targets, the position \mathbf{p}_r^τ is estimated and the velocity input \mathbf{v}_r^τ is communicated to the drones.

Note that in $\psi_{1r}(\mathbf{x})$ and $c(\mathbf{x})$, other terms involving $\sum_{i=1}^N w_{ir} \mathbf{\Pi}_{\beta_{ir}}$ which could be separated appears (see also (3.18)), but not the term $\text{vec}(\dot{\mathbf{G}}_r)^T \frac{\partial^2 \lambda_1}{\partial \text{vec}(\mathbf{G}_r)^2} \text{vec}(\dot{\mathbf{G}}_r)$, hence $\sum_{i=1}^N w_{ir} \mathbf{\Pi}_{\beta_{ir}}$ needs to be locally known. This is the reason for which these components were not written as separable, since anyway all the terms are going to be computed locally.

The implemented dynamic average consensus run by drone i to estimate quantities related to target r is in the form:

$$\begin{aligned} \dot{\mathbf{v}}_{ir}^c(t) &= -\zeta \mathbf{v}_{ir}^c(t) - K_p \sum_{j \in \mathcal{N}_i} [\mathbf{v}_{ir}^c(t) - \mathbf{v}_{jr}^c(t)] + K_i \sum_{j \in \mathcal{N}_i} [\mathbf{w}_{ir}^c(t) - \mathbf{w}_{jr}^c(t)] + \zeta \mathbf{u}_{ir}^c(t) \\ \dot{\mathbf{w}}_{ir}^c(t) &= -K_i \sum_{j \in \mathcal{N}_i} [\mathbf{v}_{ir}^c(t) - \mathbf{v}_{jr}^c(t)] \end{aligned} \quad (3.30)$$

where \mathbf{u}_{ir}^c is the input, $[\mathbf{v}_{ir}^c{}^T \mathbf{w}_{ir}^c{}^T]^T$ is the internal state of the estimator, ζ is the global estimator parameter, K_p and K_i gains. The goal of the average consensus is to converge to the average of the inputs, namely to $\frac{1}{N} \sum_{i=1}^N \mathbf{u}_{ir}^c(t)$. Thus, in this case the consensus input is set to $\mathbf{u}_{ir}^c = w_{ir} \mathbf{\Pi}_{\beta_{ir}}$ and \mathbf{v}_{ir}^c will converge to $\frac{1}{N} \sum_{i=1}^N w_{ir} \mathbf{\Pi}_{\beta_{ir}}$ and is further multiplied by N to obtain the sum of the measurements.

The minimum eigenvalue is not differentiable when it becomes not simple, i.e. with algebraic multiplicity larger than one. For this reason, in order to avoid non-smoothness of the control input, a smooth approximation of the minimum function should be implemented. In particular, we use the smooth approximation used in [43] and [37] for the r -th target

$$\bar{\lambda}_{1r} = \left(\sum_{i=1}^3 \lambda_{ir}^p \right)^{\frac{1}{p}} \quad (3.31)$$

If $p \ll 0$, then $\bar{\lambda}_{1r} \approx \lambda_{1r}(\mathbf{G}_r)$.

A straightforward implementation of (3.27) could imply that, when $L_{gi}\psi_{1r}(\mathbf{x})$ approaches zero, the QP may result infeasible. In order to solve this issue, a slack variable δ_{ir} is added, rendering the constraint a *soft* constraint.

$$\begin{aligned} \min_{\mathbf{u}_i \in \mathcal{U}_i, \delta_{ir}} \quad & \|\mathbf{u} - \mathbf{u}_d\|_2^2 + K_\delta w_{ir} \delta_{ir}^2 \\ \text{s.t.} \quad & L_{gi}\psi_{1r}(\mathbf{x})\mathbf{u}_i + d_i(\mathbf{x}) + \delta_{ir} \geq -\frac{1}{N} (c(\mathbf{x}) + \psi_{1r}(\mathbf{x})) \end{aligned} \quad (3.32)$$

Notice that the slack variable in the cost function is weighted by the product of a very high gain K_δ and the weight w_{ir} . The reason of this is two-fold: 1) to avoid numerical problems when $L_{gi}\psi_{1r}(\mathbf{x})$ is very small, which would cause problems in the multi-target case, 2) to relax the constraint for quadrotors which are not very near to the target, this in practice means that quadrotors with very small weight will ignore the constraint and track the desired input.

For the formation to track multiple targets one can simply take the intersection of the safe sets corresponding to each target, i.e. add one linear inequality constraint for each target to the QP.

$$\begin{aligned} \min_{\mathbf{u}_i \in \mathcal{U}_i, \delta_{ir}} \quad & \|\mathbf{u} - \mathbf{u}_d\|_2^2 + \sum_{r=1}^M K_\delta w_{ir} \delta_{ir}^2 \\ \text{s.t.} \quad & L_{gi}\psi_{1r}(\mathbf{x})\mathbf{u}_i + d_i(\mathbf{x}) + \delta_{ir} \geq -\frac{1}{N} (c(\mathbf{x}) + \psi_{1r}(\mathbf{x})) \\ & r = 1, \dots, M \end{aligned} \quad (3.33)$$

Note that, without the slack variables, the multitarget case could lead to infeasibility of the QP. This is because a drone could have more (possibly)

conflicting constraints (i.e. the localization of multiple targets which may move in opposite directions) and the drone is not aware of the inputs of the other drones. The use of slack variables makes some constraints soft, avoiding this situation. In general, the higher the K_δ the harder is the constraint. Since the effect of K_δ is weighted by w_{ir} , the constraint is progressively softened as w_{ir} decreases, thus allowing the i -th drone to violate the corresponding constraint. In the case of multiple targets with opposite trajectories, the drones will split in groups to observe all the targets, as will be shown in Chapter 4.

Notice that, with the proposed formulation, the quadrotors could stop following a target r if the collected information becomes high enough, because it is no more necessary to keep sensing the target and accumulate information. In this situation, the weight of each robot with respect to target r would become very small when the quadrotors start implementing the desired input. In this case, no quadrotor would then be able to reach the target again. For this reason it is added another CBF, which is used to ensure that $\sum_{i=1}^N w_{ir} \geq \gamma$ for $r = 1, \dots, M$ and $\gamma > 0$. Defining the safe set as

$$\mathcal{C}_{wr} = \{\mathbf{x} \in \mathbb{R}^{3N+(3+9)} : h_{wr} := \sum_{i=1}^N w_{ir} - \gamma \geq 0\}, \quad (3.34)$$

the additional CBF constraint can be written as:

$$\sum_{i=1}^N \frac{\partial w_{ir}}{\partial \mathbf{p}_i} \mathbf{u}_i + \sum_{i=1}^N \frac{\partial w_{ir}}{\partial \mathbf{p}_r^\tau} \mathbf{v}_r^\tau \geq - \left(\sum_{i=1}^N w_{ir} - \gamma \right) \quad (3.35)$$

where the expressions of the derivatives can be found in Appendix B. Also in this case the estimated positions and velocities are used in practice.

Remark 2. *Depending on the design of the weights, this constraint does not necessarily imply that one of the quadrotors is forced to continuously observe the target. Rather, it ensures that the quadrotors remain close enough to the target so that they can exploit the gradient information in the weights for approaching and measuring again the target whenever necessary.*

Again, this constraint can be decentralized as before, hence each drone solves

the following QP:

$$\begin{aligned}
\min_{\mathbf{u}_i \in \mathcal{U}_i, \delta_{ir}, \delta_{wir}} \quad & \|\mathbf{u} - \mathbf{u}_d\|_2^2 + \sum_{r=1}^M K_\delta w_{ir} \delta_{ir}^2 + \sum_{r=1}^M K_\delta w_{ir} \delta_{wir}^2 \\
\text{s.t.} \quad & L_{gi} \psi_{1r}(\mathbf{x}) \mathbf{u}_i + d_i(\mathbf{x}) + \delta_{ir} \geq -\frac{1}{N} (c(\mathbf{x}) + \psi_{1r}(\mathbf{x})) \\
& \frac{\partial w_{ir}}{\partial \mathbf{p}_i} \mathbf{u}_i + \frac{\partial w_{ir}}{\partial \mathbf{p}_r^\tau} \mathbf{v}_r^\tau + \delta_{wir} \geq -\frac{1}{N} \left(\sum_{i=1}^N w_{ir} - \gamma \right) \\
& r = 1, \dots, M
\end{aligned} \tag{3.36}$$

where $\frac{1}{N} \sum_{i=1}^N w_{ir}$ is obtained through average consensus (3.30) with input $\mathbf{u}_{ir}^c = w_{ir}$ and is then multiplied by N to obtain the sum of the weights.

For further convenience, it is also reported the QP (3.36) without relaxation variable and for one target

$$\begin{aligned}
\min_{\mathbf{u}_i \in \mathcal{U}_i} \quad & \|\mathbf{u} - \mathbf{u}_d\|_2^2 \\
\text{s.t.} \quad & L_{gi} \psi_{1r}(\mathbf{x}) \mathbf{u}_i + d_i(\mathbf{x}) \geq -\frac{1}{N} (c(\mathbf{x}) + \psi_{1r}(\mathbf{x})) \\
& \frac{\partial w_{ir}}{\partial \mathbf{p}_i} \mathbf{u}_i + \frac{\partial w_{ir}}{\partial \mathbf{p}_r^\tau} \mathbf{v}_r^\tau \geq -\frac{1}{N} \left(\sum_{i=1}^N w_{ir} - \gamma \right)
\end{aligned} \tag{3.37}$$

Note that (3.37) can be used safely only with one target because, since each drone does not know the input of the others, two or more hard constraints (not relaxed by δ_{ir}) could cause an unfeasible QP. This is avoided in (3.36) thanks to the relaxation variables.

Chapter 4

Simulations

In this section, the performance of the proposed solution is analyzed through numerical simulations implemented in Python, with a graphical 3D representation developed using the Matplotlib library [44]. The optimization problems are solved in the CVXPY environment [45], [46].

All the simulations were performed with $N = 6$ quadrotors, numbered from 1 to 6, and 1 or 2 targets, called in the following target 1 and target 2, respectively. The targets are assumed to be moving on a plane with constant or time-varying velocity trajectories. However, we stress that the algorithm does not require the robots to move on a plane and for testing purposes the quadrotors never use this information. The continuous-time dynamics are discretized using the forward Euler method with discretization step of 0.02 s.

For each simulation, we report the norm of the position estimation error, computed as $\|\mathbf{e}_p(t)\|_2 = \sqrt{(\mathbf{x}_r^T - \hat{\mathbf{x}}_{ir}^{\tau+})^T (\mathbf{x}_r^T - \hat{\mathbf{x}}_{ir}^{\tau+})}$, the functions ψ_0 and ψ_1 , the perception awareness weights and the Euclidean norm of the velocity control input computed by solving the QP. For the last simulation also the bearing error is shown.

Colors associated to different drones are coherent across figures, and, for the sake of clarity, are reported in Figure 4.1. Note that, in some cases, due to the consensus reached, the lines may be superimposed and thus indistinguishable.

The sensing parameters used for all the simulations are reported in the caption of Figure 3.1. Other parameters common to all the simulations are in Table 4.1, while the gains used for the dynamic consensus are in Table 4.2. The drones initial positions are uniformly sampled from a box of sizes $4 \times 4 \times 2$ m centered in $[0, 0, 3.5]^T$ m, and their velocities are limited to $\mathbf{u} \in [-3, 3]$ m/s. The initial position of target 1 and 2 are $[0.1, 0, 0]^T$ m and $[-0.1, 0, 0]^T$ m, respectively.






	drone 1
	drone 2
	drone 3
	drone 4
	drone 5
	drone 6

Figure 4.1: Colors associated to drones for the simulations.

Parameter	Symbol	Value
Forgetting factor	ρ	0.6
Minimum information level	ϵ	0.05
Threshold on sum of weights	γ	0.5
Slack variable weight	K_δ	1e6
Consensus iterations per step	K	1
Covariance matrix	\mathbf{Q}	$0.01\mathbf{I}$
Information gain	b	1

Table 4.1: Parameters common to all simulations.

Consensus on $\sum_{i=1}^N w_{ir} \mathbf{\Pi}_{\beta_{ir}}$		Consensus on $\sum_{i=1}^N w_{ir}$	
Parameter	Value	Parameter	Value
ζ	4	ζ	8
K_p	5	K_p	12
K_i	3	K_i	6

Table 4.2: Dynamic Consensus parameters.

As regards the initialisations, $\hat{\mathbf{p}}_{ir}^{\tau-}$ is initialized to a random guess, $\hat{\mathbf{v}}_{ir}^{\tau-}$ to zero, \mathbf{W}_{ir}^- to \mathbf{I} and \mathbf{G}_{ir} to $2\epsilon\mathbf{I}$.

Collision avoidance is also implemented using CBFs as in [28], but as the topology of the communication graph is fixed, it is not implemented in a distributed way. For completeness, the CBF of order 1 obtained adapting the approach in [28] is reported

$$h_{ca}(\mathbf{p}_i, \mathbf{p}_j) = \|\mathbf{p}_i - \mathbf{p}_j\|^2 - \Delta^2, \quad \forall j \in \mathcal{N}_{dist,i} \quad (4.1)$$

where Δ is the minimum allowed inter-agent distance (set to 0.5 m) and $\mathcal{N}_{dist,i}$ is the set of quadcopters considered by drone i for the collision avoidance, namely drones within a certain radius with respect to drone i . We stress again the fact that $\mathcal{N}_{dist,i} \neq \mathcal{N}_i$, hence the collision avoidance is not implemented in a distributed way. As regards the constraint to add to the

QP, since the model of the drones is a single integrator, the computations simplify and one obtains

$$L_g h_{ca}(\mathbf{p}_i, \mathbf{p}_j) \mathbf{u} + h_{ca}(\mathbf{p}_i, \mathbf{p}_j) \geq 0 \quad (4.2)$$

with $L_g h_{ca}(\mathbf{p}_i, \mathbf{p}_j) = 2(\mathbf{p}_i - \mathbf{p}_j)$. Thus, a linear inequality constraint (4.2) is added to QP (3.36) and (3.37) $\forall j \in \mathcal{N}_{dist,i}$.

In the simulations, the target(s) are initially visible by all agents, and the control is started after the information (λ_{1r}) has increased over the threshold ϵ with a dynamics dictated by the forgetting factor ρ .

The initial configuration of the simulations can be seen in Figure 4.2. Note that in the Figure only one target is present, but in case of more targets the initial positions of the drones will be the same.

In the next paragraphs, the simulations performed are discussed, showing different cases of interest.

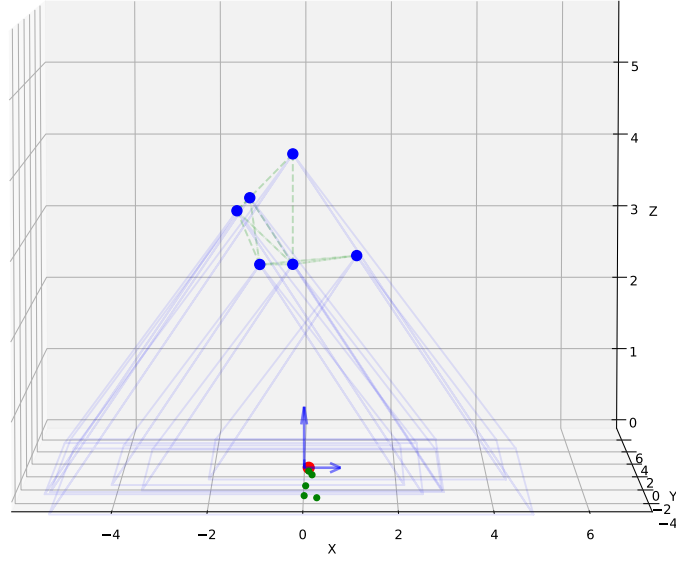


Figure 4.2: Initial configuration of the simulations at $t = 0$: drones (blue dots), target (red dot), position estimates (green dots), communication edges (green dotted lines), FoVs (blue thin lines).

4.1 Case 1: Single target with linear trajectory and no additional tasks

In this simulation, a single target moves along a linear trajectory starting from zero initial velocity and then reaching constant velocity $\mathbf{v}_1^T =$

$[0.2, 0.3, 0]^T$ m/s . The dynamics of drones, target and information are simulated for $T_{max} = 30$ s . The drones inputs are given by the solution of the QP (3.37) with $\mathbf{u}_d = [0, 0, 0]^T$, hence the constraints are hard.

The results related to this simulation are depicted in Figures 4.3, 4.4, 4.5, 4.6 and 4.7. Note that in both Figure 4.3 and Figure 4.4 are depicted $N = 6$ lines (one for each drone), but, thanks to the consensus reached, they are superimposed.

The position errors in Figure 4.3 converges to 0, meaning that the estimated state of the target converges to the true one. The behavior of the position errors between $t = 0$ s and $t = 4$ s is due to the fact that the state estimate is initially similar to the true one, but as the target starts moving with a non-zero velocity the estimation worsen until the estimated target velocity converges to the true one.

In Figure 4.4, ψ_{01} and ψ_{11} , after the initial accumulation of information due to the fact that target is visible by all agents at initial time, decrease and when they approach 0 the HOCBF constraint of the QP becomes active, forcing the quadcopters to move to maintain the global information above the threshold ϵ , namely the CBF positive. Indeed, from the control inputs in Figure 4.6, it is evident that the QP produces an input different from 0 only when the CBF vanishes approaching 0. The inputs present an initial peak that depends on the relative distance of drone i to target 1, and then all the inputs converges to the velocity of the target, meaning that they are collectively following it. Note, indeed, that the final value of the norm of the control inputs converges to $\sqrt{0.2^2 + 0.3^2} = 0.36$ m/s , which is the (constant) norm of the velocity of target 1.

Another interesting behavior emerging from this simulation, visible from Figure 4.7, is the tendency of the drones to autonomously split in 2 groups to sense the target from 2 different and approximately symmetric directions. This is in agreement with the PE considerations about bearing made by in [31]. From the same Figure, one can also note that the drones follow the target moving initially upward and then stabilizing at constant height. It is easy to understand that, to maintain a target in the FoV, a drone can go upward, go in the same direction of the target or mix these behaviors as in this case. A similar behavior is present also in the next two simulations.

Finally, the perception awareness weights for this simulation are reported in Figure 4.5, from which we can appreciate that all the weights start from 1 and then, as the target moves outside the FoV, they decrease until the HOCBF constraint becomes active, finally stabilizing to constant values. Note, however, that each drone has a low weight relative to target 1 ($w_{i1} < 0.2$), meaning that each drone brings a poor information contribution to the system, and hence all of them are necessary to respect the constraint. This Figure suggests that not all drones should be necessary to satisfy the localization constraints. This situation can be avoided with the use of the slack variables, as analyzed in the next simulation, where also the

multi-target case is tackled.

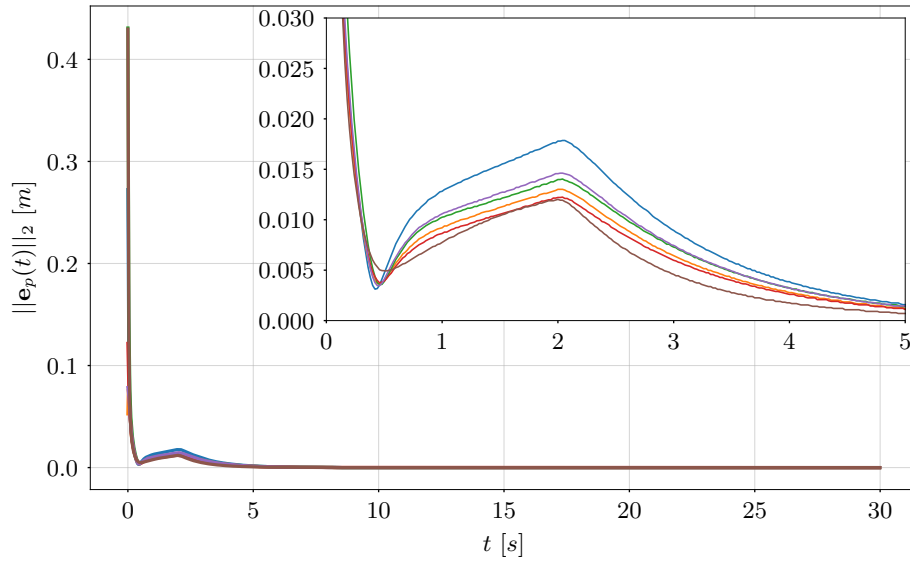


Figure 4.3: Case 1: position errors.

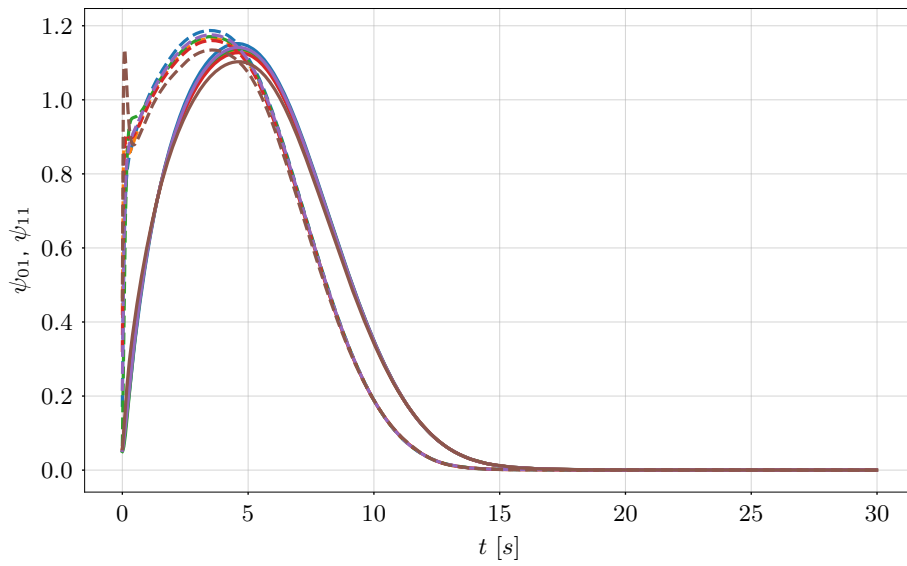


Figure 4.4: Case 1: ψ_{01} (solid), ψ_{11} (dotted).

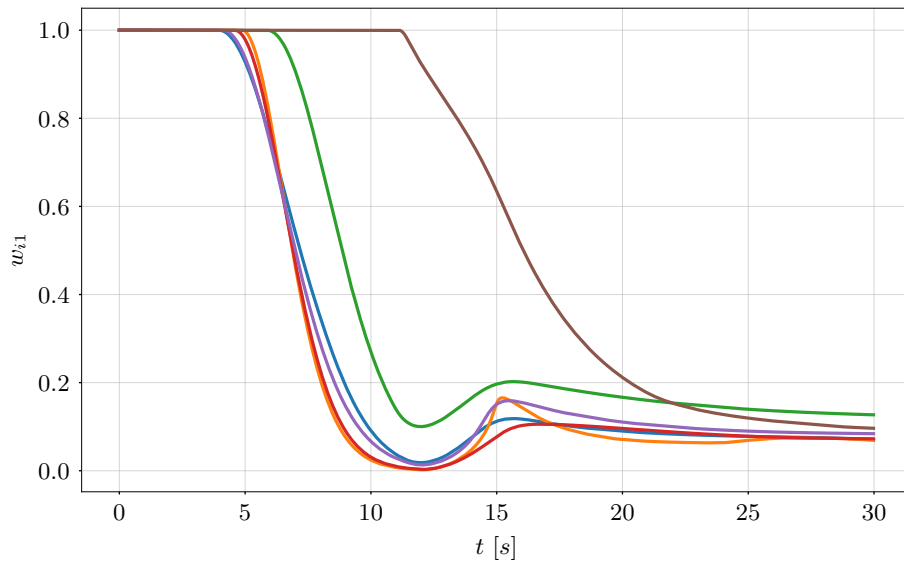


Figure 4.5: Case 1: weights.

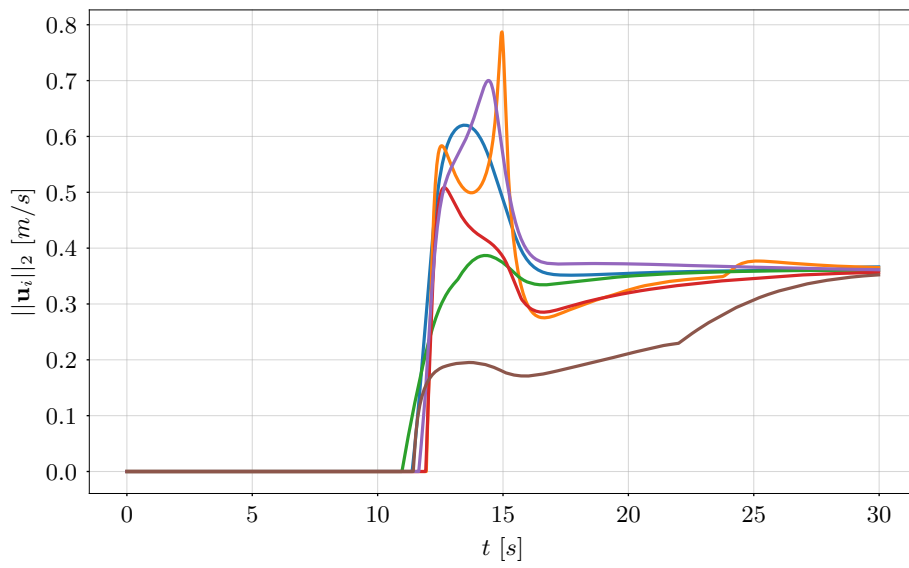


Figure 4.6: Case 1: norm of control inputs.

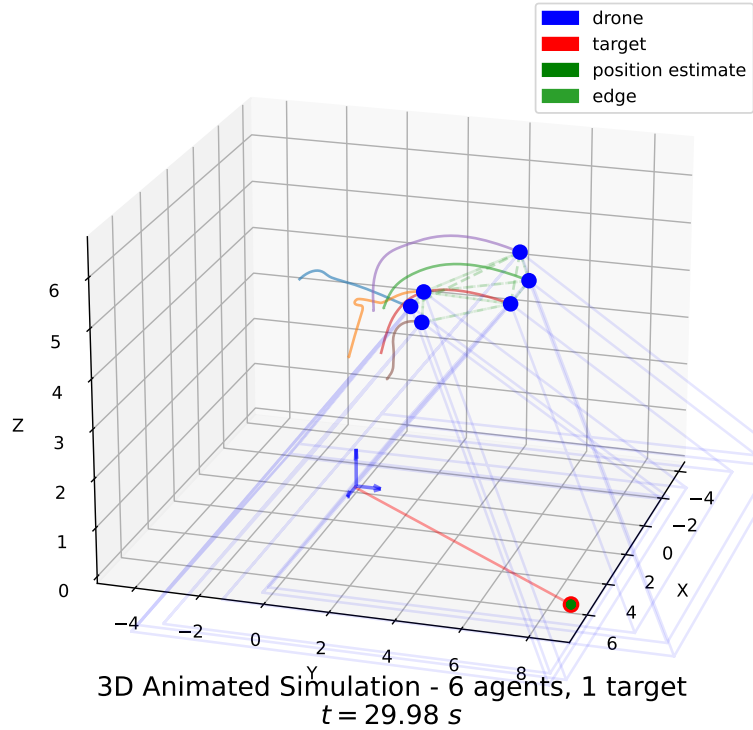


Figure 4.7: Case 1: trajectories at the end of the simulation. The blue thin lines represent the FoV of the drones. The drones split in two clearly visible groups.

4.2 Case 2: Multi-target with sinusoidal trajectory and no additional tasks

In this simulation, 2 targets are present. Target 1 moves with the same trajectory of the previous simulation, while target 2 moves with a planar sinusoidal trajectory with amplitude 2.2 m and pulse 0.15 rad/s , hence with time-varying velocity. This simulation has a duration of $T_{max} = 50 \text{ s}$. Each drone solve a QP (3.36) with $\mathbf{u}_d = [0, 0, 0]^T$. The results are reported in Figures 4.8, 4.9, 4.10, 4.11 and 4.12. The upper plots are relative to target 1, while the lower plots are related to target 2.

In Figure 4.8, it is shown that, in the linear trajectory case, the error converges to zero, while in the time-varying velocity case the error remains bounded, as expected from Chapter 2.

For both targets, the HOCBFs constraints are satisfied, as depicted in Figure 4.9. From Figure 4.10, looking at the weights different from 0 after the transient, one can appreciate that only drone 3 (green lines) and drone 6 (brown lines) moves to localize target 1, and analogously drone 1 (blue) and

5 (purple) localize target 2. This behaviour can also be noted by looking at the inputs shapes from Figure 4.11, in which can be distinguished the drones which are following the target with the linear trajectory from the ones which are following the target with sinusoidal trajectory.

As in the previous simulation, the drones observe the targets from 2 symmetric directions, as shown in Figure 4.12, which depicts drones and targets at the final instant of the simulation. It is important to notice that, thanks to the use of the relaxation variables weighted by w_{ir} , the localization constraints of drones 2 and 4 are made soft, hence they can be violated resulting in zero drone input. In Figure 4.12, they are depicted with an X to indicate that they are not sensing either of the two targets. These drones are not necessary for the satisfaction of the constraints and could thus perform other tasks. Despite this, the thresholds on the minimum eigenvalues of the OGs are satisfied. This result confirms that not all 6 drones are necessary to maintain the localization constraints, since only 2 are sufficient for each target.

There is the possibility that ψ_{01} and ψ_{11} go slightly negative during the transient. There are two reasons for this: first, some minor approximations are done in the computation of the derivatives, and hence the model is not perfect; secondly, the slack variables are added to *all* the localization constraints and thus they are not hard, even if some are weighted by weights similar to 1. The effect of weighting K_δ with w_{ir} is that the drones far from the j -th target will have a very small weight and thus a constraint much softer than the drones that are actively sensing the target.

4.3 Case 3: Single target with linear trajectory and bearing-only formation control

In this simulation, only one target is present and moves with constant velocity along a linear trajectory (starting smoothly as in the first simulation) until $t = 25$ s, then it stops and moves backward with opposite constant velocity toward the starting position. The total duration of this simulation is $T_{max} = 45$ s. Each drones solve the QP (3.36) with \mathbf{u}_d obtained from a bearing-only formation control law [47]. The desired formation is a planar circle. The bearing-only control law only fixes directions and not distances, hence the formation is able to expand and contract maintaining the directions defined by the desired bearings \mathbf{g}_{ij}^* . The simulation results are depicted in Figures 4.13, 4.14, 4.15, 4.16 and 4.18. Figure 4.17 shows the error on the formation bearings, computed as $\|\mathbf{e}_\beta\|_2 = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \|\mathbf{g}_{ij} - \mathbf{g}_{ij}^*\|_2$. The formation expands as 2 drones are following the target, then, when the target comes back to the starting position, its motion becomes compatible with the localization constraint and hence the QP stops altering the controller velocity \mathbf{u}_d .

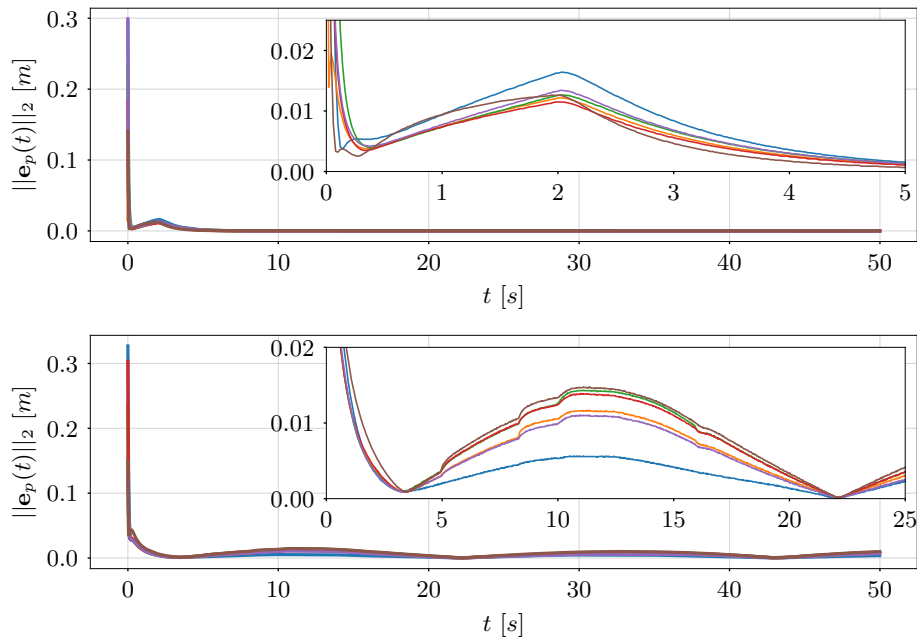


Figure 4.8: Case 2: position errors.

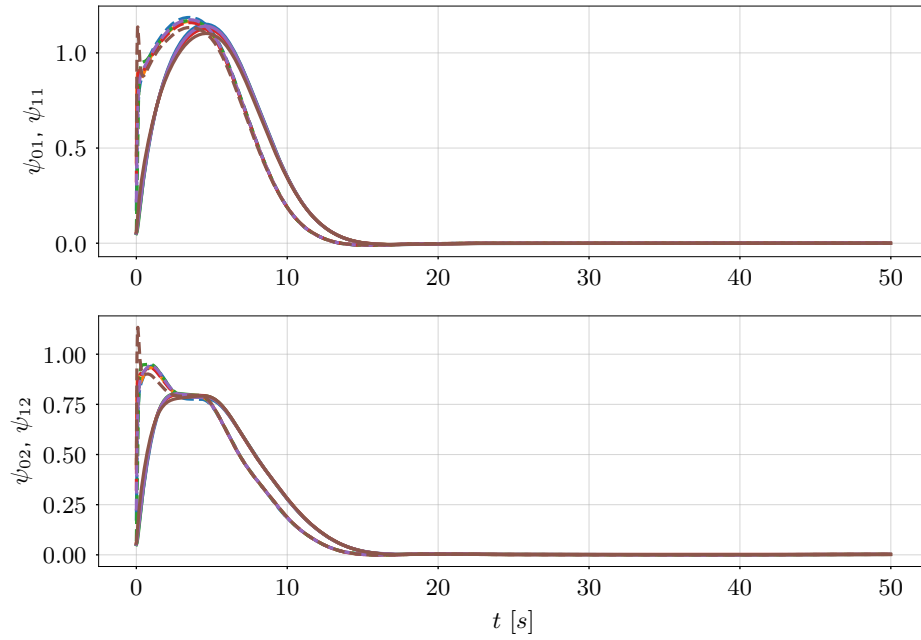
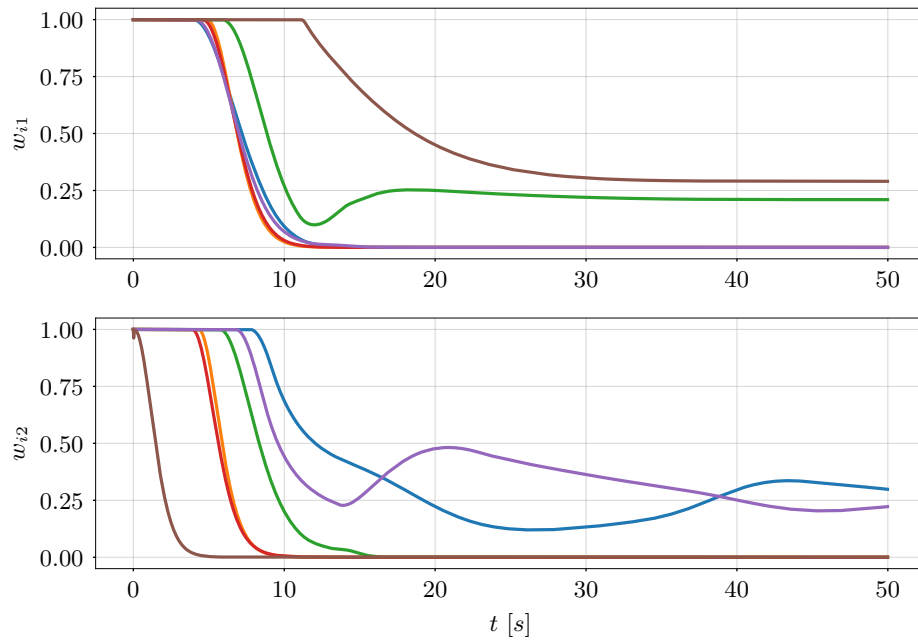
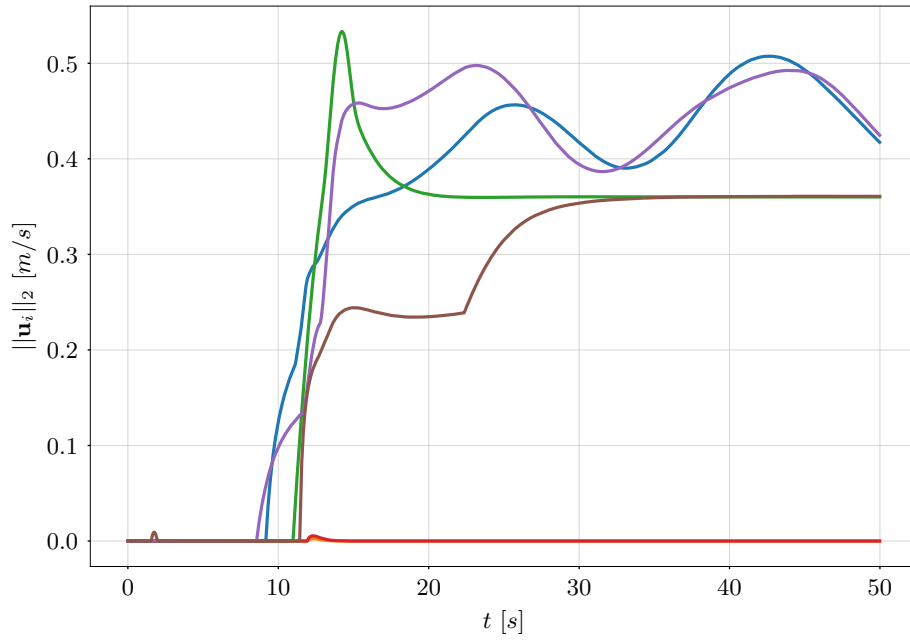


Figure 4.9: Case 2: ψ_0 (solid), ψ_1 (dotted).

**Figure 4.10:** Case 2: weights.**Figure 4.11:** Case 2: norm of control inputs.

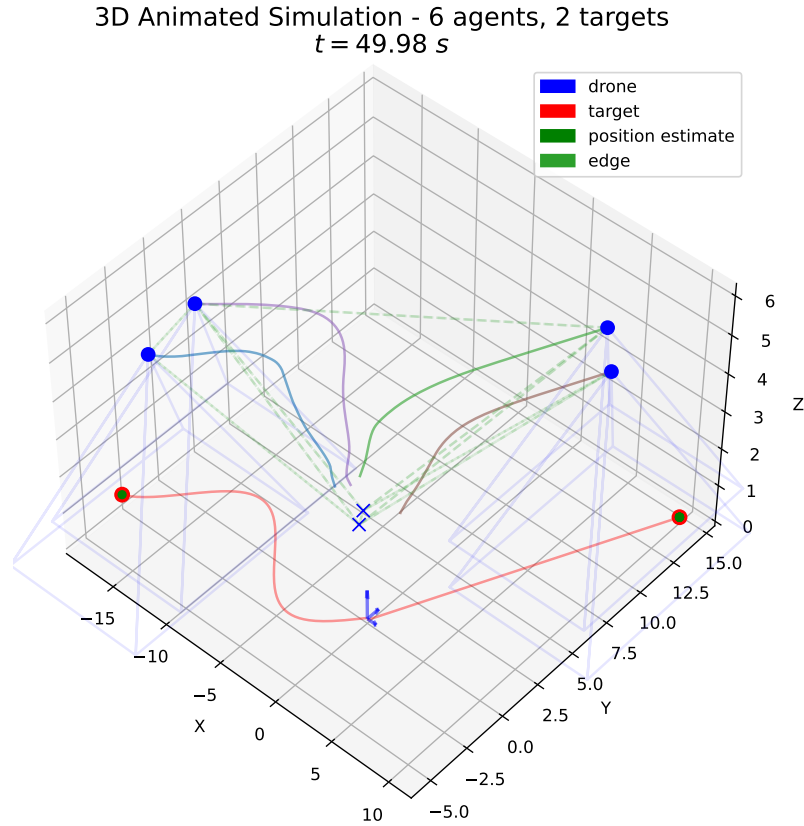


Figure 4.12: Case 2: trajectories at the end of the simulation. The blue thin lines represent the FoV of the drones. Each target is followed by two drones from two approximately symmetric directions.

From Figure 4.13 one can see that the error temporarily increases when the target start moving in the opposite direction, because the velocity of the target is non-constant for a time interval. In Figure 4.14, it can be understood when the HOCBF constraint is active and when the information increases due to the target going toward the start position, where it can be observed by more drones. Looking at Figure 4.15 it is clear that only drones 1, 2 and to a lesser extent also 3 moves to observe the target and thus have a weight different from 0, as confirmed by Figure 4.16. In the inputs plot, one can also see that there is an initial peak due to formation control law, when this input is not modified by the QP because the target is visible by all agents and thus the HOCBF constraint is satisfied. Then, the 3 above mentioned drones follow the target as ψ_{01} ψ_{11} approaches zero and the others move to maintain the desired directions, expanding the formation. Finally, the input converges to zero when the target comes back to the original position. This

behavior is clear also from Figure 4.16, since only the drones that participates to the localization have a bearing error significantly higher than zero, because their formation control input is altered to satisfy the localization constraints. Figure 4.18 depicts the trajectories and positions of drones and target at the final instant of the simulation. As can be seen, the planar formation is clearly formed and the target is visible by all the quadcopters, which are free to implement the control \mathbf{u}_d .

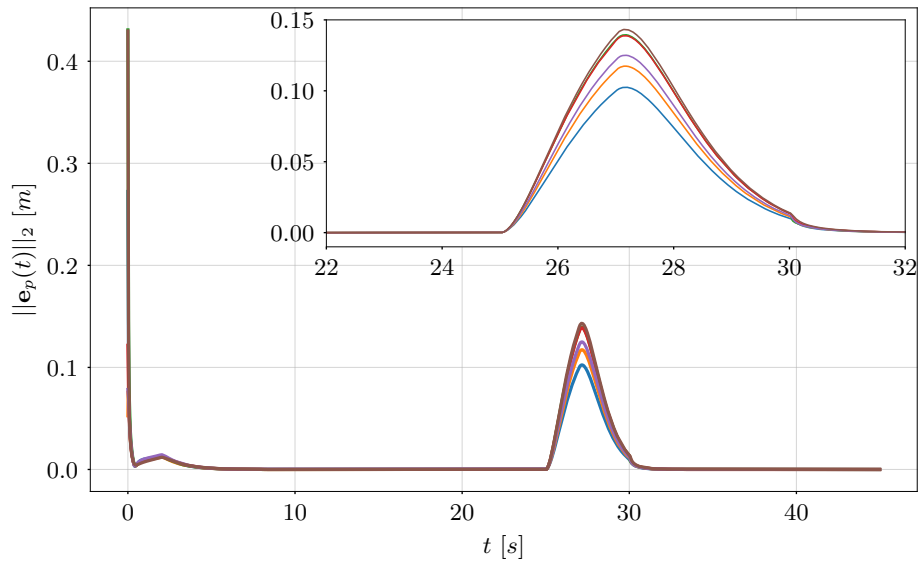


Figure 4.13: Case 3: position errors.

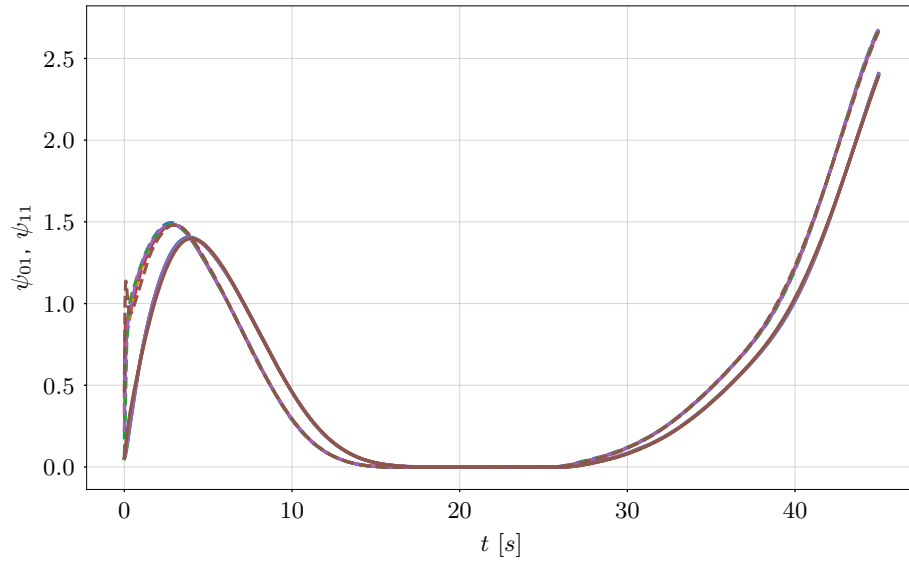


Figure 4.14: Case 3: ψ_{01} (solid), ψ_{11} (dotted).

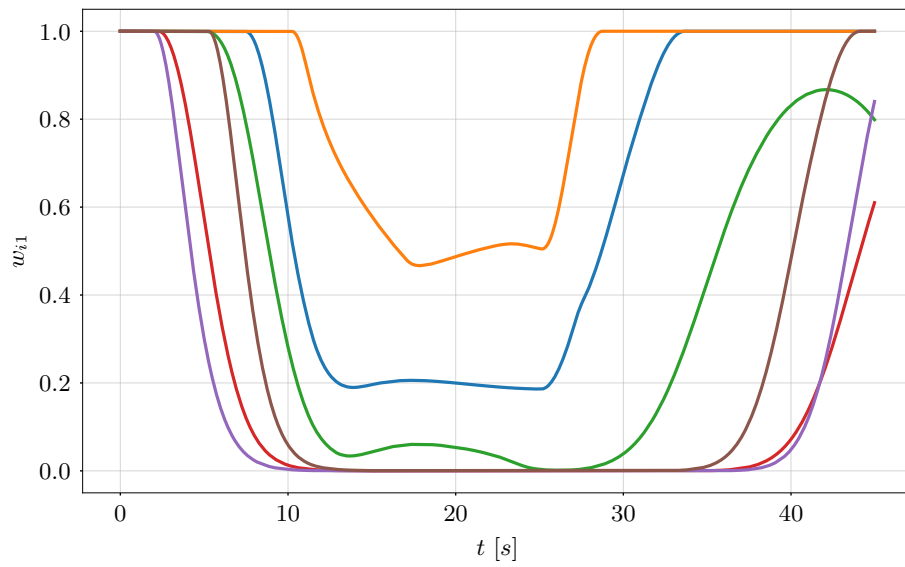


Figure 4.15: Case 3: weights.

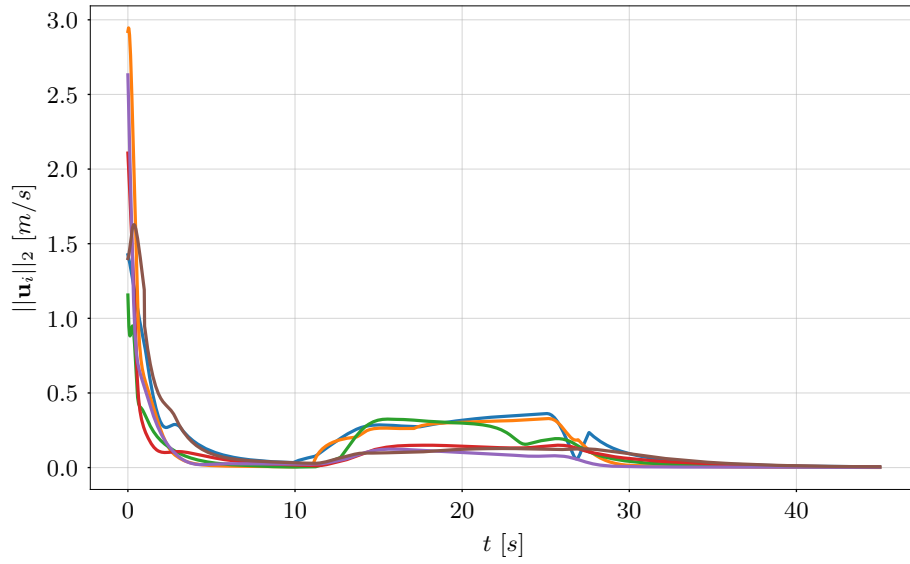


Figure 4.16: Case 3: norm of control inputs.

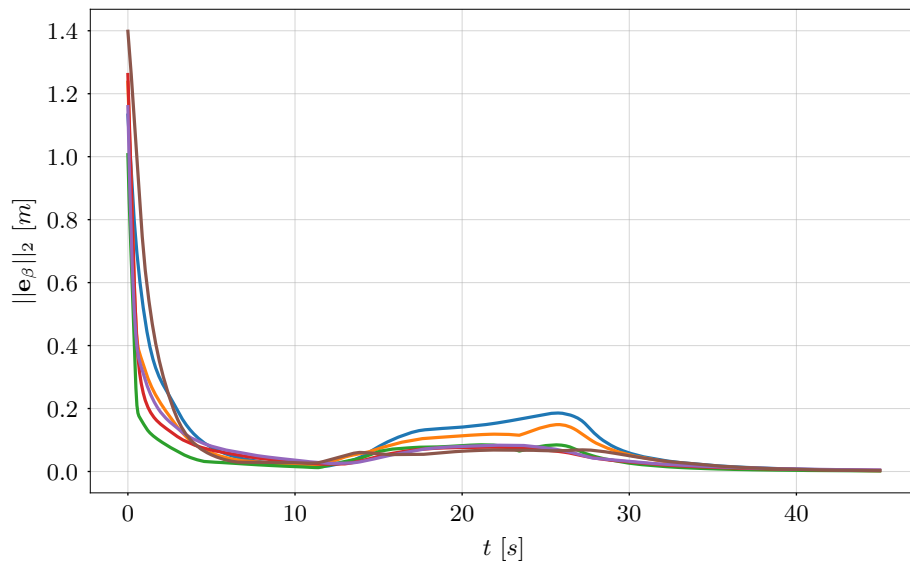


Figure 4.17: Case 3: bearing errors.

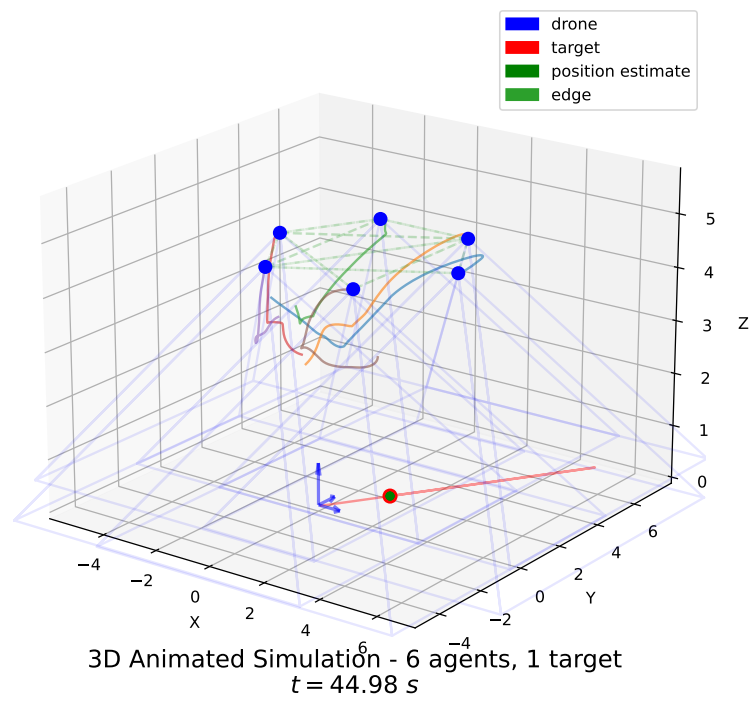


Figure 4.18: Case 3: trajectories at the end of the simulation. The blue thin lines represent the FoV of the drones. The drones are in planar circular formation.

Chapter 5

Conclusions and Future Works

In this Thesis, a distributed persistent monitoring scheme for estimating the state of one or multiple moving target(s) from bearing measurements by employing an Information Consensus Filter is proposed and analyzed. The filter relies on a standard Information Consensus Filter (ICF), modified to work with relative bearing measurements, and is uniformly globally exponentially stable under Persistency of Excitation (PE) conditions. The main contribution of this work is to guarantee that the PE conditions are met also in presence of sensing constraints. This is achieved by relying on two main tools: the decentralized High Order Control Barrier Functions, used to enforce the invariance of the safe set, and the weighted Observability Gramian with forgetting factor, which is used to quantify the persistency of excitation. The centralized solution is presented first, and then it is decentralized in order to respect the centralized constraints. The proposed approach can deal with multiple targets thanks to a proper relaxation of the constraints. Finally, the proposed approach is validated through numerical simulations.

In future, it will be of interest to consider a time-varying graph topology with connectivity maintenance and to extend this approach to ensure localizability of formations from relative measures.

Appendix A

Matrix calculus rules

For further convenience and in order to clarify the notation used in Chapter 3 and Appendix B, some useful matrix calculus rules are reported in this Appendix.

If $f(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$ is a function of $\mathbf{x} \in \mathbb{R}^n$, its derivative with respect to \mathbf{x} is a row vector and its gradient is a column vector

$$\frac{\partial f}{\partial \mathbf{x}} = \left[\frac{\partial f}{\partial x_1} \quad \cdots \quad \frac{\partial f}{\partial x_n} \right] = \nabla f^T \quad (\text{A.1})$$

If $\mathbf{f}(\mathbf{x})$ is a vector-valued function $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, its derivative with respect to \mathbf{x} is a matrix

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{m \times n} \quad (\text{A.2})$$

As a rule, we can state that when we differentiate a function, the result has as many rows as the dimension of the function (m) and as many columns as the dimension of the variable with respect to we are differentiating (n). Moreover, given two scalar functions $u(\mathbf{x}), v(\mathbf{x})$, the derivative of their product with respect to \mathbf{x} is

$$\frac{\partial uv}{\partial \mathbf{x}} = u \frac{\partial v}{\partial \mathbf{x}} + v \frac{\partial u}{\partial \mathbf{x}} \quad (\text{A.3})$$

Given a scalar function $v(\mathbf{x})$ and a vector function $\mathbf{u}(\mathbf{x})$ we have

$$\frac{\partial v\mathbf{u}}{\partial \mathbf{x}} = v \frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \mathbf{u} \frac{\partial v}{\partial \mathbf{x}} \quad (\text{A.4})$$

Finally, given two vector functions $\mathbf{u}(\mathbf{x})$ and $\mathbf{v}(\mathbf{x})$, we have

$$\frac{\partial(\mathbf{u} \cdot \mathbf{v})}{\partial \mathbf{x}} = \frac{\partial \mathbf{u}^\top \mathbf{v}}{\partial \mathbf{x}} = \mathbf{u}^\top \frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \quad (\text{A.5})$$

Appendix B

Computations

In this Appendix are reported all the computations associated with the HOCBFs in Chapter 3.

For convenience, define the number $d = 3N + (3 + 9)$ used for the dimension of the state. Some useful quantities, already defined in previous chapters, are reported first. The variables involved have already been explained in Chapter 3.

Safe set:

$$\mathcal{C}_r = \{\mathbf{x} \in \mathbb{R}^d : h_r(\mathbf{x}) = \lambda_{1r}(\mathbf{x}) - \epsilon \geq 0\} \quad (\text{B.1})$$

State of the system:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{p}_r^T \\ \text{vec}(\mathbf{G}_r) \end{bmatrix} \in \mathbb{R}^d \quad (\text{B.2})$$

Functions appearing in the state dynamics $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t) + \mathbf{g}(\mathbf{x})\mathbf{u}$:

$$\mathbf{f}(\mathbf{x}, t) = \begin{bmatrix} \mathbf{0} \\ \mathbf{v}_r^T(t) \\ \text{vec}\left(-\rho\mathbf{G}_r + \sum_{i=1}^N bw_{ir}\mathbf{\Pi}_{\beta_{ir}}\right) \end{bmatrix} \in \mathbb{R}^d \quad (\text{B.3})$$

$$\mathbf{g}(\mathbf{x}) = \begin{bmatrix} (\mathbf{1}_N \otimes \mathbf{I}_3) \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \in \mathbb{R}^{d \times 3N} \quad (\text{B.4})$$

Functions ψ :

$$\begin{aligned} \psi_{0r}(\mathbf{x}) &= h_r(\mathbf{x}) = \lambda_{1r}(\mathbf{x}) - \epsilon \\ \psi_{1r}(\mathbf{x}) &= L_f(h_r(\mathbf{x})) = \frac{\partial \lambda_{1r}(\mathbf{x})}{\partial(\text{vec}(\mathbf{G}_r))} \text{vec}(\dot{\mathbf{G}}_r) + (\lambda_{1r} - \epsilon) \end{aligned} \quad (\text{B.5})$$

Recall that in the formal definition of the state the true target position is used, but in practice it is not available to the drones and hence its estimate $\hat{\mathbf{p}}_{ir}^{\tau+}$ is used. The same holds for the target velocity.

B.1 Computation of $L_f(h_r(\mathbf{x}))$

The first Lie derivative explained is $L_f(h_r(\mathbf{x}))$ in (B.5).

$$\begin{aligned}
L_f h_r(\mathbf{x}) &= L_f \lambda_{1r}(\mathbf{x}) = \underbrace{\frac{\partial \lambda_{1r}(\mathbf{x})}{\partial \mathbf{x}}}_{1 \times d} \underbrace{\mathbf{f}(\mathbf{x})}_{d \times 1} = \\
&= \underbrace{\left[\frac{\partial \lambda_{1r}}{\partial \mathbf{p}} \quad \frac{\partial \lambda_{1r}}{\partial \mathbf{p}_r^T} \quad \frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \right]}_{\substack{1 \times 3N & 1 \times 3 & 1 \times 9}} \begin{bmatrix} \mathbf{0}_{3N \times 1} \\ \mathbf{v}_r^T_{3 \times 1} \\ \text{vec}(\dot{\mathbf{G}}_r)_{9 \times 1} \end{bmatrix} = \quad (\text{B.6}) \\
&= \frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \in \mathbb{R}
\end{aligned}$$

The terms $\frac{\partial \lambda_{1r}}{\partial \mathbf{p}}$ and $\frac{\partial \lambda_{1r}}{\partial \mathbf{p}_r^T}$ are zero as the minimum eigenvalue of the OG does not depend instantaneously by the positions of drone and target, while the term $\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)}$ can be computed knowing the eigenvector \mathbf{v}_{1r} associated to eigenvalue λ_{1r} , using Theorem 1 of [42], which holds for symmetric matrices and normalized eigenvectors

$$\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} = \mathbf{v}_{1r}^T \otimes \mathbf{v}_{1r}^T \quad \text{or} \quad \frac{\partial \lambda_{1r}}{\partial \mathbf{G}_r} = \mathbf{v}_{1r} \mathbf{v}_{1r}^T \quad (\text{B.7})$$

where the symbol \otimes denotes the Kronecker product.

In practice, for control purposes, a smooth approximation of λ_{1r} is used, and $\frac{\partial \bar{\lambda}}{\partial \text{vec}(\mathbf{G}_r)}$ can be computed applying the chain rule and recalling that $\bar{\lambda} = \bar{\lambda}(\lambda_1, \lambda_2, \lambda_3)$

$$\frac{\partial \bar{\lambda}}{\partial \text{vec}(\mathbf{G}_r)} = \sum_{i=1}^3 \frac{\partial \bar{\lambda}}{\partial \lambda_{ir}} \frac{\partial \lambda_{ir}}{\partial \text{vec}(\mathbf{G}_r)} = \left(\sum_{i=1}^3 \lambda_{ir}^p \right)^{\frac{1}{p}-1} \left(\sum_{i=1}^3 \lambda_{ir}^{p-1} \frac{\partial \lambda_{ir}}{\partial \text{vec}(\mathbf{G}_r)} \right) \quad (\text{B.8})$$

B.2 Computation of $L_{g_i} \psi_{1r}(\mathbf{x})$

It is now explained the computation of $L_{g_i} \psi_{1r}(\mathbf{x})$, where

$$\mathbf{g}_i(\mathbf{x}) = \begin{bmatrix} \mathbf{I}_{3 \times 3} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \in \mathbb{R}^{15 \times 3} \quad i = 1, \dots, N \quad (\text{B.9})$$

$$\begin{aligned}
L_{gi}\psi_{1r}(\mathbf{x}) &= L_{gi}L_f\lambda_{1r}(\mathbf{x}) = \frac{\partial}{\partial \mathbf{x}_i} (L_f\lambda_{1r}) \mathbf{g}_i(\mathbf{x}) = \frac{\partial}{\partial \mathbf{x}_i} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) \mathbf{g}_i(\mathbf{x}) = \\
&= \underbrace{\left[\frac{\partial}{\partial \mathbf{p}_i} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) \right]}_{1 \times 3} \underbrace{\left[\frac{\partial}{\partial \mathbf{p}_r^T} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) \right]}_{1 \times 3} \underbrace{\left[\frac{\partial}{\partial \text{vec}(\mathbf{G}_r)} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) \right]}_{1 \times 9} \underbrace{\mathbf{g}_i(\mathbf{x})}_{15 \times 1}
\end{aligned} \tag{B.10}$$

Due to the particular structure of \mathbf{g}_i , only the first 1×3 term of (B.10) is not multiplied by zero. The first 1×3 block of the expression above which multiplies $\mathbf{g}_i(\mathbf{x})$ can be expanded following the rules in (A.5)

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{p}_i} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) &= \\
&= \underbrace{\text{vec}(\dot{\mathbf{G}}_r)^T \frac{\partial}{\partial \mathbf{p}_i} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \right)}_{=0} + \frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \frac{\partial}{\partial \mathbf{p}_i} \left(\text{vec}(\dot{\mathbf{G}}_r) \right)
\end{aligned} \tag{B.11}$$

The first term of expression (B.11) is zero, because the derivative of the eigenvector with respect to the vectorized matrix doesn't depend instantaneously on the positions. Therefore, the final expression of $L_{gi}\psi_{1r}(\mathbf{x})$ reads as

$$L_{gi}\psi_{1r}(\mathbf{x}) = \underbrace{\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)}}_{1 \times 9} \underbrace{\frac{\partial \text{vec}(\dot{\mathbf{G}}_r)}{\partial \mathbf{p}_i}}_{9 \times 3} \in \mathbb{R}^{1 \times 3} \tag{B.12}$$

Note that the dimension is consistent because this term will be multiplied by $\mathbf{u}_i \in \mathbb{R}^{3 \times 1}$ to obtain a scalar.

In (B.12), the term $\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)}$ has already been computed in (B.7) and in (B.8) considering the smooth approximation of the eigenvalue, while the term $\frac{\partial \text{vec}(\dot{\mathbf{G}}_r)}{\partial \mathbf{p}_i}$ can be computed as

$$\begin{aligned}
\frac{\partial \text{vec}(\dot{\mathbf{G}}_r)}{\partial \mathbf{p}_i} &= \frac{\partial}{\partial \mathbf{p}_i} \text{vec} \left(-\rho \mathbf{G}_r + \sum_{i=1}^N b w_{ir} \mathbf{\Pi}_{\beta_{ir}} \right) = \\
&= b w_{ir} \frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial \mathbf{p}_i} + b \text{vec}(\mathbf{\Pi}_{\beta_{ir}}) \frac{\partial w_{ir}}{\partial \mathbf{p}_i}
\end{aligned} \tag{B.13}$$

where we have considered that the OG depends on all the previous positions and thus the derivative with respect to the last position only is null, while for the other two terms the rule (A.4) has been applied. More precisely

$$\frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial \mathbf{p}_i} = \left[\frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial x_i} \quad \frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial y_i} \quad \frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial z_i} \right] \in \mathbb{R}^{9 \times 3} \tag{B.14}$$

where x_i, y_i, z_i are the components of \mathbf{p}_i . The columns of this matrix have the following structure in the non-vectorized version (the subscript of β_{ir} is omitted for brevity)

$$\begin{aligned} \frac{\partial \mathbf{\Pi}_\beta}{\partial x_i} &= \frac{\partial}{\partial x_i} (\mathbf{I} - \beta \beta^T) = -\frac{\partial \beta}{\partial x_i} \beta^T - \beta \frac{\partial \beta^T}{\partial x_i} = \\ &= -\frac{1}{d_{ir}} (-\mathbf{e}_1 + \beta^x \beta) \beta^T - \frac{1}{d_{ir}} \beta (-\mathbf{e}_1 + \beta^x \beta)^T \end{aligned} \quad (\text{B.15})$$

where $\mathbf{e}_1 = [1, 0, 0]^T$ and β^x represents the x -component of β . The expressions of the derivatives of $\mathbf{\Pi}_\beta$ with respect to y_i and z_i are similar.

To compute $\frac{\partial w_{ir}}{\partial \mathbf{p}_i}$ we use (A.3)

$$\frac{\partial w_{ir}}{\partial \mathbf{p}_i} = \frac{\partial (w_{Dir} w_{Bir})}{\partial \mathbf{p}_i} = w_{Dir} \frac{\partial w_{Bir}}{\partial \mathbf{p}_i} + w_{Bir} \frac{\partial w_{Dir}}{\partial \mathbf{p}_i} \quad (\text{B.16})$$

with

$$\frac{\partial w_{Dir}}{\partial \mathbf{p}_i} = \frac{\partial w_{Dir}}{\partial \hat{d}_{ir}} \frac{\partial \hat{d}_{ir}}{\partial \mathbf{p}_i} = -2 \frac{(\hat{d}_{ir} - D)}{\sigma_D^2} e^{-\frac{(\hat{d}_{ir} - D)^2}{\sigma_D^2}} \left(\frac{-(\hat{\mathbf{p}}_r^{\tau+} - \mathbf{p}_i)}{\hat{d}_{ir}} \right)^T \quad (\text{B.17})$$

and

$$\frac{\partial w_{Bir}}{\partial \mathbf{p}_i} = \frac{\partial w_{Bir}}{\partial \beta_{ir}} \frac{\partial \beta_{ir}}{\partial \mathbf{p}_i} = 2 \frac{(c_{ir} - \cos(\alpha_M^{th}))}{\sigma_B^2} e^{-\frac{(c_{ir} - \cos(\alpha_M^{th}))^2}{\sigma_B^2}} \left(\frac{-\mathbf{\Pi}_{\beta_{ir}}}{\hat{d}_{ir}} \right) \quad (\text{B.18})$$

where rules for the differentiation of unit vectors and norms have been applied. In (B.17), D and σ_D^2 should be substituted with D_m^{th} and $\sigma_{D_m}^2$ or D_M^{th} and $\sigma_{D_M}^2$, following the definition of the weights provided in Chapter 3.

The expressions above hold when the r -th target approaches the perception limits and the weights depends on the position, otherwise they are constant and their derivative are zero. Note that the last term in (B.17) is $-\beta_{ir}^T$.

Substituting the expressions above in (B.12), equation (3.22) is obtained. This concludes the computation of $L_{gi} \psi_{1r}(\mathbf{x})$.

It is important remember that the used positions and velocities of the targets are in practice the *estimated* quantities $\hat{\mathbf{p}}_{ir}^{\tau+}$ and $\hat{\mathbf{v}}_{ir}^{\tau+}$, since the real ones are not available to the drones.

B.3 Computation of $L_f \psi_{1r}(\mathbf{x})$

The last derivative needed for the constraints of the QP is $L_f \psi_{1r}(\mathbf{x})$.

$$L_f \psi_{1r}(\mathbf{x}) = L_f^2 h_r(\mathbf{x}) + L_f h_r(\mathbf{x}) \in \mathbb{R} \quad (\text{B.19})$$

The term $L_f h_r(\mathbf{x})$ has already been computed, while for the term $L_f^2 h_r(\mathbf{x})$ it holds

$$L_f^2 h_r(\mathbf{x}) = L_f(L_f \lambda_{1r}(\mathbf{x})) = \frac{\partial}{\partial \mathbf{x}} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) \mathbf{f}(\mathbf{x}) =$$

$$\underbrace{\left[\frac{\partial}{\partial \mathbf{p}} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) \right]}_{1 \times 3N} \underbrace{\frac{\partial}{\partial \mathbf{p}_r^\tau} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right)}_{1 \times 3} \underbrace{\frac{\partial}{\partial \text{vec}(\mathbf{G}_r)} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right)}_{1 \times 9} \underbrace{\mathbf{f}(\mathbf{x})}_{d \times 1} \quad (\text{B.20})$$

Due to the particular structure of $\mathbf{f}(\mathbf{z})$, the first $1 \times 3N$ part is multiplied by zeros and does not contribute to the final result. The second term of (B.20) is

$$\frac{\partial}{\partial \mathbf{p}_r^\tau} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) =$$

$$= \underbrace{\text{vec}(\dot{\mathbf{G}}_r)^T \frac{\partial}{\partial \mathbf{p}_r^\tau} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \right)}_{=0} + \frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \frac{\partial}{\partial \mathbf{p}_r^\tau} \left(\text{vec}(\dot{\mathbf{G}}_r) \right) \quad (\text{B.21})$$

where

$$\frac{\partial \text{vec}(\dot{\mathbf{G}}_r)}{\partial \mathbf{p}_r^\tau} = \sum_{i=1}^N \left(b w_{ir} \frac{\partial \text{vec}(\mathbf{\Pi}_{\beta_{ir}})}{\partial \mathbf{p}_r^\tau} + b \text{vec}(\mathbf{\Pi}_{\beta_{ir}}) \frac{\partial w_{ir}}{\partial \mathbf{p}_r^\tau} \right) = \sum_{i=1}^N \left(-\frac{\partial \text{vec}(\dot{\mathbf{G}}_r)}{\partial \mathbf{p}_i} \right) \quad (\text{B.22})$$

The third term of (B.20) is

$$\frac{\partial}{\partial \text{vec}(\mathbf{G}_r)} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \text{vec}(\dot{\mathbf{G}}_r) \right) =$$

$$= \frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \frac{\partial \text{vec}(\dot{\mathbf{G}}_r)}{\partial \text{vec}(\mathbf{G}_r)} + \text{vec}(\dot{\mathbf{G}}_r)^T \frac{\partial}{\partial \text{vec}(\mathbf{G}_r)} \left(\frac{\partial \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r)} \right) \quad (\text{B.23})$$

where

$$\frac{\partial \text{vec}(\dot{\mathbf{G}}_r)}{\partial \text{vec}(\mathbf{G}_r)} = -\rho \mathbf{I}_{9 \times 9} \quad (\text{B.24})$$

and using Theorem 4 of [42]

$$\frac{\partial^2 \lambda_{1r}}{\partial \text{vec}(\mathbf{G}_r) \partial (\text{vec}(\mathbf{G}_r))^T} = \mathbf{K}_3 \left(\mathbf{Y}_{1r}^\dagger \otimes \mathbf{v}_{1r} \mathbf{v}_{1r}^T + \mathbf{v}_{1r} \mathbf{v}_{1r}^T \otimes \mathbf{Y}_{1r}^\dagger \right) \quad (\text{B.25})$$

where \mathbf{K}_3 is the 9×9 *commutation matrix*, used to transform a vectorized matrix in the vectorized version of its transpose, and \mathbf{Y}_{1r}^\dagger is the pseudo inverse of $\mathbf{Y}_{1r} = \lambda_{1r} \mathbf{I} - \mathbf{G}_r$.

In (B.24) we consider that the projection matrix that appears in the dynamics of the information matrix does not depend on the information matrix itself.

Again, remember that the positions and velocities of the targets used in practice are the *estimated* quantities $\hat{\mathbf{p}}_{ir}^{\tau+}$ and $\hat{\mathbf{v}}_{ir}^{\tau+}$.

As before, in practice a smooth approximation is used for the minimum eigenvalue, hence

$$\begin{aligned} & \frac{\partial^2 \bar{\lambda}}{\partial \text{vec}(\mathbf{G}_r) \partial (\text{vec}(\mathbf{G}_r))^T} = \\ & = \sum_{i=1}^3 \left(\frac{\partial \bar{\lambda}}{\partial \lambda_{ir}} \frac{\partial^2 \lambda_{ir}}{\partial \text{vec}(\mathbf{G}_r) \partial (\text{vec}(\mathbf{G}_r))^T} + \frac{\partial \lambda_{ir}}{\partial \text{vec}(\mathbf{G}_r)} \frac{\partial}{\partial \text{vec}(\mathbf{G}_r)} \left(\frac{\partial \bar{\lambda}}{\partial \lambda_{ir}} \right) \right) \end{aligned} \quad (\text{B.26})$$

with

$$\frac{\partial \bar{\lambda}}{\partial \lambda_{ir}} = \frac{\partial}{\partial \lambda_{ir}} \left(\sum_{i=1}^3 \lambda_{ir}^p \right)^{\frac{1}{p}} = \left(\sum_{i=1}^3 \lambda_{ir}^p \right)^{\frac{1}{p}-1} \left(\sum_{i=1}^3 \lambda_{ir}^{p-1} \right) \quad (\text{B.27})$$

and

$$\frac{\partial}{\partial \text{vec}(\mathbf{G}_r)} \left(\frac{\partial \bar{\lambda}}{\partial \lambda_{ir}} \right) = \frac{\partial^2 \bar{\lambda}}{\partial \lambda_{ir}^2} \frac{\partial \lambda_{ir}}{\partial \text{vec}(\mathbf{G}_r)} \quad (\text{B.28})$$

where

$$\frac{\partial^2 \bar{\lambda}}{\partial \lambda_{ir}^2} = (1-p) \left(\sum_{i=1}^3 \lambda_{ir}^p \right)^{\frac{1-2p}{p}} \left(\sum_{i=1}^3 \lambda_{ir}^{p-1} \right)^2 + (p-1) \left(\sum_{i=1}^3 \lambda_{ir}^p \right)^{\frac{1-p}{p}} \left(\sum_{i=1}^3 \lambda_{ir}^{p-2} \right) \quad (\text{B.29})$$

Substituting the equations above in (B.19) we obtain equation (3.23).

Acknowledgements

I sincerely thank the entire Rainbow Team of the Inria/IRISA center in Rennes, Brittany, France, where I spent six months for the development of this Thesis. In particular, special thanks are for Nicola De Carli and Paolo Robuffo Giordano, who wisely guided me along my path.

It was an important experience for my growth from both personal and professional points of view. I hope that our routes will cross again, inside or outside the Engineering world.

Bibliography

- [1] R. Olfati-Saber, “Kalman-consensus filter: Optimality, stability, and performance,” in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009, pp. 7036–7042.
- [2] A. T. Kamal, J. A. Farrell, and A. K. Roy-Chowdhury, “Information weighted consensus filters and their application in distributed camera networks,” *IEEE Transactions on Automatic Control*, vol. 58, no. 12, pp. 3112–3125, 2013.
- [3] G. Battistelli and L. Chisci, “Kullback–leibler average, consensus on probability densities, and distributed state estimation with guaranteed stability,” *Automatica*, vol. 50, no. 3, pp. 707–718, 2014.
- [4] A. T. Kamal, J. A. Farrell, and A. K. Roy-Chowdhury, “Information weighted consensus,” in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, IEEE, 2012, pp. 2732–2737.
- [5] P. Yang, R. A. Freeman, and K. M. Lynch, “Distributed cooperative active sensing using consensus filters,” in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, IEEE, 2007, pp. 405–410.
- [6] F. Morbidi and G. L. Mariottini, “Active target tracking and cooperative localization for teams of aerial vehicles,” *IEEE transactions on control systems technology*, vol. 21, no. 5, pp. 1694–1707, 2012.
- [7] S. Martínez and F. Bullo, “Optimal sensor placement and motion coordination for target tracking,” *Automatica*, vol. 42, no. 4, pp. 661–668, 2006.
- [8] R. Olfati-Saber, “Distributed tracking for mobile sensor networks with information-driven mobility,” in *2007 American Control Conference*, IEEE, 2007, pp. 4606–4612.
- [9] R. Olfati-Saber and P. Jalalkamali, “Coupled distributed estimation and control for mobile sensor networks,” *IEEE Transactions on Automatic Control*, vol. 57, no. 10, pp. 2609–2614, 2012.
- [10] C. Freundlich, S. Lee, and M. M. Zavlanos, “Distributed active state estimation with user-specified accuracy,” *IEEE Transactions on Automatic Control*, vol. 63, no. 2, pp. 418–433, 2017.

- [11] A. J. Krener and K. Ide, “Measures of unobservability,” in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009, pp. 6401–6406.
- [12] M. Jacquet, M. Kivits, H. Das, and A. Franchi, “Motor-level n-mpc for cooperative active perception with multiple heterogeneous uavs,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2063–2070, 2022.
- [13] R. A. Freeman, P. Yang, and K. M. Lynch, “Stability and convergence properties of dynamic average consensus estimators,” in *Proceedings of the 45th IEEE Conference on Decision and Control*, IEEE, 2006, pp. 338–343.
- [14] S. S. Kia, B. Van Scoy, J. Cortes, R. A. Freeman, K. M. Lynch, and S. Martinez, “Tutorial on dynamic average consensus: The problem, its applications, and the algorithms,” *IEEE Control Systems Magazine*, vol. 39, no. 3, pp. 40–72, 2019.
- [15] L. Luft, T. Schubert, S. I. Roumeliotis, and W. Burgard, “Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication,” *The International Journal of Robotics Research*, vol. 37, no. 10, pp. 1152–1167, 2018.
- [16] S.-J. Chung, A. A. Paranjape, P. Dames, S. Shen, and V. Kumar, “A survey on aerial swarm robotics,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 837–855, 2018.
- [17] N. Nigam and I. Kroo, “Persistent surveillance using multiple unmanned air vehicles,” in *2008 IEEE Aerospace Conference*, IEEE, 2008, pp. 1–14.
- [18] N. Nigam, S. Bieniawski, I. Kroo, and J. Vian, “Control of multiple uavs for persistent surveillance: Algorithm and flight test results,” *IEEE Transactions on Control Systems Technology*, vol. 20, no. 5, pp. 1236–1251, 2011.
- [19] N. Michael, E. Stump, and K. Mohta, “Persistent surveillance with a team of mavs,” in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2011, pp. 2708–2714.
- [20] S. K. K. Hari, S. Rathinam, S. Darbha, K. Kalyanam, S. G. Manyam, and D. Casbeer, “The generalized persistent monitoring problem,” in *2019 American Control Conference (ACC)*, IEEE, 2019, pp. 2783–2788.
- [21] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” in *2019 18th European control conference (ECC)*, IEEE, 2019, pp. 3420–3431.
- [22] K. P. Tee, S. S. Ge, and E. H. Tay, “Barrier lyapunov functions for the control of output-constrained nonlinear systems,” *Automatica*, vol. 45, no. 4, pp. 918–927, 2009.

- [23] X. Xu, P. Tabuada, J. W. Grizzle, and A. D. Ames, “Robustness of control barrier functions for safety critical control,” *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 54–61, 2015.
- [24] Q. Nguyen and K. Sreenath, “Exponential control barrier functions for enforcing high relative-degree safety-critical constraints,” in *2016 American Control Conference (ACC)*, IEEE, 2016, pp. 322–328.
- [25] W. Xiao and C. Belta, “Control barrier functions for systems with high relative degree,” in *2019 IEEE 58th conference on decision and control (CDC)*, IEEE, 2019, pp. 474–479.
- [26] W. Xiao and C. Belta, “High order control barrier functions,” *IEEE Transactions on Automatic Control*, 2021.
- [27] X. Tan, W. S. Cortez, and D. V. Dimarogonas, “High-order barrier functions: Robustness, safety, and performance-critical control,” *IEEE Transactions on Automatic Control*, vol. 67, no. 6, pp. 3021–3028, 2021.
- [28] L. Wang, A. D. Ames, and M. Egerstedt, “Safety barrier certificates for collisions-free multirobot systems,” *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017.
- [29] B. Capelli and L. Sabattini, “Connectivity maintenance: Global and optimized approach through control barrier functions,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 5590–5596.
- [30] L. Lindemann and D. V. Dimarogonas, “Barrier function based collaborative control of multiple robots under signal temporal logic tasks,” *IEEE Transactions on Control of Network Systems*, vol. 7, no. 4, pp. 1916–1928, 2020.
- [31] T. Hamel and C. Samson, “Riccati observers for position and velocity bias estimation from direction measurements,” in *2016 IEEE 55th conference on decision and control (CDC)*, IEEE, 2016, pp. 2047–2053.
- [32] F. Garin and L. Schenato, “A survey on distributed estimation and control applications using linear consensus algorithms,” *Networked control systems*, pp. 75–107, 2010.
- [33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
- [34] M. Ruiz, A. Vargas, and V. Romero-Cano, “Detection and tracking of a landing platform for aerial robotics applications,” Nov. 2018, pp. 1–6. DOI: 10.1109/CCRA.2018.8588112.
- [35] F. Bullo, *Lectures on network systems*. Kindle Direct Publishing, 2020, vol. 1.
- [36] B. T. Hinson and K. A. Morgansen, “Observability optimization for the nonholonomic integrator,” in *2013 American Control Conference*, IEEE, 2013, pp. 4257–4262.

- [37] P. Salaris, M. Cagnetti, R. Spica, and P. R. Giordano, “Online optimal perception-aware trajectory generation,” *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1307–1322, 2019.
- [38] F. Pukelsheim, *Optimal design of experiments*. SIAM, 2006.
- [39] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
- [40] P. Bernard, N. Mimmo, and L. Marconi, “On the semi-global stability of an ek-like filter,” *IEEE Control Systems Letters*, vol. 5, no. 5, pp. 1771–1776, 2020.
- [41] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs for safety critical systems,” *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.
- [42] J. R. Magnus, “On differentiating eigenvalues and eigenvectors,” *Econometric Theory*, vol. 1, pp. 179–191, 1985.
- [43] R. Spica and P. Robuffo Giordano, “Active decentralized scale estimation for bearing-based localization,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 5084–5091.
- [44] J. D. Hunter, “Matplotlib: A 2d graphics environment,” *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. DOI: 10.1109/MCSE.2007.55.
- [45] S. Diamond and S. Boyd, “CVXPY: A Python-embedded modeling language for convex optimization,” *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.
- [46] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, “A rewriting system for convex optimization problems,” *Journal of Control and Decision*, vol. 5, no. 1, pp. 42–60, 2018.
- [47] S. Zhao and D. Zelazo, “Bearing rigidity and almost global bearing-only formation stabilization,” *IEEE Transactions on Automatic Control*, vol. 61, no. 5, pp. 1255–1268, 2015.