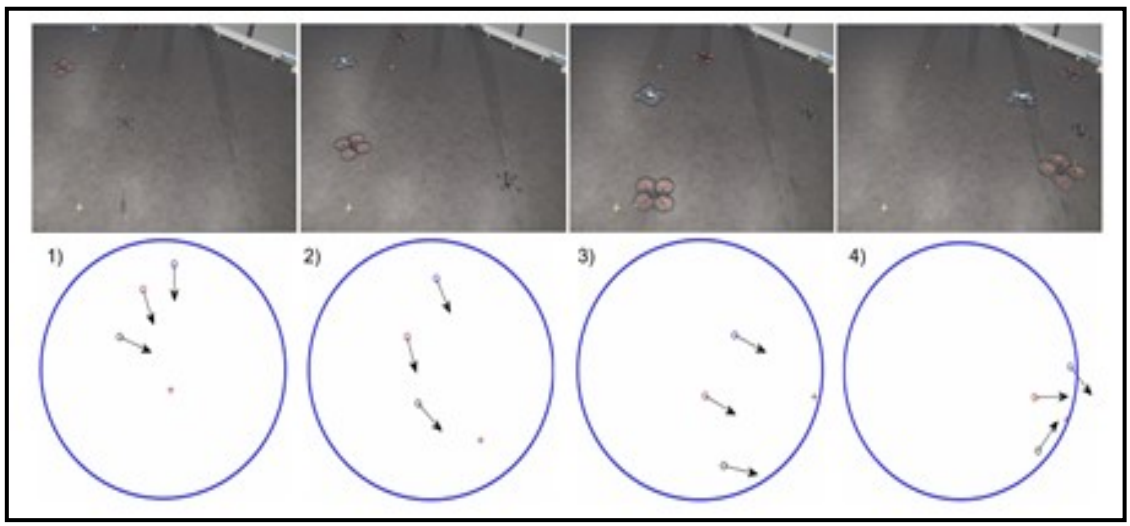
Hunter drones : drones cooperation for tracking an intruder drone

Cristino de Souza Junior

Par **Cristino DE SOUZA JUNIOR**

Hunter drones : drones cooperation for tracking an intruder drone

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 19 avril 2021
Spécialité : Automatique et Robotique : Unité de recherche Heudyasic (UMR-7253)
D2608

Université de Technologie de Compiègne
Heudiasyc

Doctoral School **Rue du docteur Schweitzer CS 60319**

University Department **Laboratoire Heudiasyc UMR 7253**

Thesis defended by **Cristino de Souza Junior**

Defended on **19th April, 2021**

In order to become Doctor from Université de Technologie de Compiègne and from Heudiasyc

Spécialité : Automatique et Robotique

Hunter drones

Drones cooperation for tracking an intruder drone

Thesis supervised by	Pedro Castillo	Supervisor
	Borislav Vidolov	Co-Supervisor

Committee members

Referees	Nicolas Bredeche	Professor at ISIR
	Olivier Simonin	Professor at INSA Lyon
Examiners	Eliseo Ferrante	Associate Professor at VU
	Amal Elfallah	Professor at LIP6
	Cristina Maniu	CentraleSupélec/L2S
	Phillippe Bonnifait	HDS
Supervisors	Pedro Castillo	CNRS
	Borislav Vidolov	Associate Professor at UTC

Colophon

Doctoral dissertation entitled “Hunter drones”, written by Cristino de Souza Junior, completed on 20th June, 2021, typeset with the document preparation system LaTeX and the yathesis class dedicated to theses prepared in France.

Hunter drones
Drones cooperation for tracking an intruder drone

Abstract

In the last decade, we have witnessed significant advances in multi-robot systems. Much attention has been paid to the modeling of coordinated movement, called flocking; however, some current applications require more than the ability to navigate cohesively and without collision, for example, the use of drones to intercept a faster intruder. This application, to which this thesis is dedicated, although it is a subset of collective motion, has characteristics contrary to flocking, such as dispersion, in place of aggregation, and capture, instead of keeping desired distance.
To reproduce more efficient and performing group-pursuit behavior in robotic applications, we propose in this work several behavior-based multi-agent strategies. Along most of this work, the interaction rules between the pursuers and the target are designed based on the geometric rules and relative kinematic models, commonly used in missiles' guidance laws.
Furthermore, we investigate the application of the proposed strategies in real-time robots. For that, a multilayer motion controller architecture is proposed, allowing the application of high-level navigation laws in a quadcopter.
Our strategy shares concepts with classical multi-agents methods [49, 77, 58], such as local sense, limited interaction, and decentralization decision-make. Nevertheless, we differ from most of them for considering the environment's perception of polar and relative coordinates, which is a consistent assumption considering embedded sensors (LIDAR and camera).
Besides, our work extends known techniques of navigation guidance laws to the multi-agent problem. In other words, pursuer behavior is given by modified guidance laws, where the parameters are adapted according to the pursuit's engagement. As a result, we have emergent group ambush with non-crossing trajectories between pursuers.

Keywords: group pursuit, multiagents, hunting, drones, flocking.

Laboratoire Heudiasyc UMR 7253
Batiment Blaise Pascal – 67, Avenue de Landshut – CS 60319 – 60203 Compiègne Cedex – France

Hunter drones
Drones cooperation for tracking an intruder drone

Résumé

Au cours de la dernière décennie, nous avons assisté à des avancées significatives dans les systèmes multi-robots. Une grande attention a été portée à la modélisation du mouvement coordonné, appelé flocage ; cependant, certaines applications actuelles nécessitent plus qu'une navigation cohérente ; comme par exemple, l'utilisation de drones pour intercepter un intrus plus rapide. Cette application, bien qu'elle soit un sous-ensemble du mouvement collectif, présente des caractéristiques contraires au flocage, telles que la dispersion, au lieu de l'agrégation, et la capture, au lieu de garder la distance souhaitée.
Pour reproduire un comportement de poursuite de groupe plus efficace, nous proposons dans ce travail plusieurs stratégies multi-agents basées sur le comportement. Dans la plupart de ces travaux, les règles d'interaction entre les poursuivants et la cible sont conçues à partir des règles géométriques et des modèles cinématiques relatives, couramment utilisés dans les lois de guidage des missiles. De plus, nous étudions leurs applications avec des robots en temps réel. Pour cela, une architecture de contrôle de mouvement multicouche est proposée.
Notre stratégie partage des concepts avec des méthodes multi-agents classiques, telles que le sens local, l'interaction limitée et décentralisation. Néanmoins, nous différons de la plupart d'entre eux pour considérer la perception par l'environnement des coordonnées polaires et relatives. Par ailleurs, nos travaux étendent les techniques connues de lois de guidage de navigation au problème multi-agents.

Mots clés : group pursuit, multiagents, hunting, drones, flocking.

Publications

Accepted Papers:

[i] C. de Souza Jr, R. Newburry, A. Cosgun, P. Castillo, B. Vidolov, D. Kulić. *Distributed Multi-Agent Pursuit using Deep Reinforcement Learning*. RAL/ICRA 2021.

[ii] C. de Souza Jr, P. Castillo, B. Vidolov, R. Lozano. *Enhanced UAV pose estimation using a KF: experimental validation*. International Conference on Unmanned Aircraft Systems, Dallas - US, 2018.

[iii] C. de Souza Jr, P. Castillo, B. Vidolov. *Reactive drone pursuit and obstacle avoidance based in parallel navigation*. 23rd IEEE International Conference on Intelligent Transportation System, 2020.

[iv] D. Andrea, C. de Souza Jr, P. Castillo, B. Vidolov. *Trajectory generation and tracking for phugoid maneuvers using a mini-airplane*. Mediterranean Conference in Control and Automation, St Raphael - France, 2020.

Papers Under Review:

[v] C. de Souza Jr, P. Castillo, B. Vidolov. *Local interaction and navigation guidance for hunters drones*. Robotica, 2021. Under the second review.

Papers in Preparation:

[vi] C. de Souza Jr, P. Castillo, B. Vidolov. *Autonomous drone pursuit using a fleet of drones with target prediction* .

[vii] C. de Souza Jr, P. Castillo, B. Vidolov. *Flocking, pursuit and avoidance: a multi-agents approach based in guidance laws*.

Contents

chi ben comincia è giá a metá
dell'opera

Italian proverb

Introduction

Motivation In the last decades, we have seen an abrupt popularization of mobile robots around the world. Consecutively, civil authorities have been facing new challenges, such as the mitigation of criminal robots. In fact, unauthorized drone flights over protected areas, such as over airports and prisons, have been common in newspaper headlines.

Although there are preventive solutions, the current technologies have serious drawbacks, such as high-cost structure required, side effects or inefficiency. Against this backdrop, an interesting solution has arisen lately: the use of anti-drone drones, which has proved itself an efficient and less costly solution. A more detailed discussion about the drone threat and the anti-drone solutions will be discussed in Section 1.1.

However, the solution discussed in this document aims at a scenario in the near future in the anti-drone fight, where the intruders can act autonomously, reactively, and can have superior capabilities than the single pursuer. This new scenario would require a group effort to accomplish the capture. The group of robots must perform more than a flight in formation or maintain the group's cohesion. They should be able to navigate cooperatively to ambush the most agile and reactive invaders, such as a group of predators behaves in real life to chase their prey.

Collective motion Many efforts have been made in the last decades to analyze and reproduce complex group behavior. Initiatives from different science areas have been launched to observe, describe, and reproduced group natural phenomena.

Although countless examples could be listed here, we left to the reader three special examples to introduce our case: first, a flock of hundreds of starlings in a complex and fluid movement, splitting in the presence of predators and smoothly joining again, producing an amazing aerial display. Second, the perfect synchronization and symmetry of a jets fleet in an airshow, with a breathtaking maneuver in very high velocities.

Moreover, finally, a wolf-pack's wit and coordination to take down a stronger or faster prey, like a bison or a hare. In this last case, the struggle for survival from both parts provides these amazing spectacles. The preys adopt unpredictable escape trajectories, and the pursuers execute well-coordinated teamwork strategies to ambush and capture their prey.

Figure 1 – Examples of collective motions: a) Flocking of starlings b) An aerial demonstration squadron c) Wolves chasing a hare

Although all of the above examples are part of the collective motion fields, they have all distinctive features. While planes' fleets have symmetry and fixed formations, the flock of birds shows incredible cohesion and a very fluid formation. In turn, the wolf-pack, although it has similarities to both, represents a completely new branch, where the individuals do not seem to care much about the formation or even cohesion of the group. Instead, they trace their own trajectory towards the prey and adapt it in the neighbors' presence, which results in a surprising emergent ambush behavior.

Multi-robots The first two examples of Figure 1, flocking and formations, have been extensively studied in the literature, and even successful applications in real robots have also been made. A flock of robots, like in [53, 80] and amazing fleet formation, like in [81, 82], are becoming common in technology fairs and entertainment shows.

Therefore, let us states the distinction between the two definitions: on the one hand, the term "flocking" refers to agent-based and decentralized approaches, where the individuals are homogeneous, having no explicitly hierarchy and able to sense and act locally. It is widespread in natural phenomena. Interacting units end up moving with about the same absolute velocity. This approach in multi-robots applications can also be referenced as "behavioral-based," where each agent acts based on a set of rules, or laws, such in [53, 58].

On the other hand, by fleet formation, we refer to all the multi-agent strategies that keep a specific pattern (constellation). It can be referred to as virtual structures strategies in multi robots, which normalize centralized and leader-following strategies. Each agent tries to keep a specific relative positioning towards the neighbors and the leader (physical or virtual). Examples of this kind of implementation can be seen in [81, 83, 141].

Both techniques, fleet formation and flocking, have been applied to the group chase evasion problem [35, 46, 47, 51, 84]. Nevertheless, there are several differences between these distinct problems; and the obtained results fall short to reproduce the collective hunting phenomena. Formation based strategies, such as in [46, 47, 84], does not differ much from the case of a single pursuer. The constellation of robots behaves as a single agent, once their movements are constrained by the formation. Similarly, flocking-based strategies, such as in [35, 51], although does not require fixed formation, are still much worried about group's cohesion, which goes contrary to spreading out behavior of hunters. The alignment (or viscous term), frequently used in flocking, actually limits the action of the individual pursuer and, once more, preventing the necessary detachment necessary for the ambushing.

These strategies mentioned above do not perform an efficient and realistic pursuit because they were not designed for this, but rather adapted from flocking or formation algorithms.

Group pursuit-evasion One last topic, which could not be left out of this introductory chapter, is the recent field of group pursuit-evasion. It aims to describe prey-predators' complex behavior considering agent-based models, such as in [34, 51, 52].

Moreover, as pointed by [85], this recent and opened field inherits from two consolidated literature: *pursuit-evasion* and *collective motion*. Firstly, it can be seen as an extension of the single pursuit-evasion problem, taking advantage of vast literature involving: analytical solutions of pursuit, differential games, and missiles guidance studies. Furthermore, it can be considered as extension of the collective motion, taking advantage of the extensive literature in self-propelled particles, such as [77], which has been used to model numerous complex systems, like bacteria, insect, physical particles, etc.

Recent work in escape from group chase [52] proposed modified versions of Vicsek's self-propelled particles to adapt the conventional flocking algorithm into a group pursuit. It was done by adding an extra chase term and modifying the alignment one. Although interesting behaviors can be exhibited in the simulations and efficient ambush behavior appears during the chase, their model has many variables, which made more difficult tunning and required optimization.

Furthermore, its applicability in the real world is limited due to its assumptions in observation; it relies on the knowledge of position and velocities of pursuers and targets expressed in the global frame. Although these assumptions can be partially afforded by considering communications between pursuers, the evader is not cooperative in real-world applications. Its global position and velocity are normally not available for the pursuer.

Thesis proposition: To reproduce more efficient and performing group-pursuit behavior in robotic applications, we propose several behavior-based multi-agent strategies. Along most of this work, the interaction rules between the pursuers and the target are designed based on the geometric rules and relative kinematic models, commonly used in missiles' guidance laws.

Furthermore, the application of these group pursuit strategies in robots in real-time will be studied in this thesis. For that, a multilayer motion controller architecture is proposed, allowing the application of high-level navigation laws in a quadcopter.

Our strategy share concepts with classical multi-agents methods [49, 58, 77], such as local sense, limited interaction, and decentralization decision-make. Nevertheless, we differ from most of them for considering the environment's perception of polar and relative coordinates, which is a consistent assumption considering embedded sensors (LIDAR and camera).

Besides, our work extends known techniques of navigation guidance laws to the multi-agent problem. In other words, pursuer behavior is given by modified guidance laws, where the parameters are adapted according to the pursuit's engagement. As a result, we have emergent group ambush with non-crossing trajectories between pursuers.

Outline of the thesis: In Chapter 1, we review essential literature related to group-pursuit applied to drones. Firstly, we go deeper into the counter-drone problem and the current solutions. Then, we present an overview of the collective motion literature, highlighting behavioral-based approaches and collective robots. After that, we introduce an overview of the pursuit-evasion problem, emphasizing navigation guidance laws and reinforcement learning. Finally, we introduce the emerging field of group pursuit-evasion, and we highlight the recent works.

In Chapter 2, we study the scenario where a group of pursuer and a single evader are traveling in a bounded 2-dimensional workspace. To solve the capture problem, we propose three distinct behavioral-based strategies for the pursuer team. The first one is based on the *Deviated Pure Pursuit* guidance, DPP, and the different values of the offset angle for each pursuer gives them different behavior in the stalking. The second strategy is an improvement of the previous one, where a *Parallel Navigation* term is added to the DPP. This allows the pursuer to take more efficient paths toward the target and still deviate from each other, avoiding collisions.

Finally, we propose a Deep Reinforcement Learning framework to obtain through simulations close-to-optimal policies for the pursuers.

In Chapter 3, slightly different scenarios or assumptions are studied. Firstly, we will assume that the target's global positioning and velocity are available for the pursuers. With that, a predictor scheme is chosen to foresee the target's future position. Secondly, we consider a scenario where just one pursuer realizes the target's stalking; nevertheless, other non-cooperative agents, or mobile obstacles, are traveling in the same workspace. In a third and last scenario, we are interested in larger amounts of agents, where not all of them are informed about the target. In this, we investigate the flocking formation considering only the relative and polar information about the neighbors.

In Chapter 4, we investigate the application of the proposed strategies into a group of quadcopters. First, we describe the experimental setup, detailing the material used for the real-time flights. Then, we describe our work on state estimation, in which we implemented a Kalman filter algorithm for filtering and estimating position and velocity. Finally, we present the motion control architecture for a drone pursuer.

Finally, we propose a careful discussion about the thesis's proposed strategies, where criteria such as performance and viability of implementation will be taken into account. Besides, we will give a general conclusion and the research prospect of this work.

The best way to defeat a tank is with another tank.

Popular saying

Chapter 1

State of the art

In this state-of-the-art, we review the essential literature related to group-pursuit applied to drones. First, we present an overview of the counter drones problem and its current solutions in the civil environment. We evidence the promising solution of anti-drone drone (ADD), and we argue the need to use a group of ADD's to contain the coming threats. Then, we delve into the problem of multi-agents, giving an overview of some key works in the literature on collective motion and a particular focus on multi-robots in the pursuit. After that, we present an overview of the classical pursuit-evasion problem, emphasizing the emerging field of group chase-evasion. Moreover, we summarize the well-established literature in guidance laws, giving theoretical concepts for its development in this work.

Contents

1.1 The counter-drone fight

The recent boom in the production and marketing of drones has profoundly affected our society. As reported in [126], the commercial use of drones has a vast potential, and they have shown themselves as a versatile tool already being used for mapping, public security, humanitarian aid, among others.

However, their illness is also a reality that has dramatically concerned the population as a whole [60]. For example, in [125], the authors point out that drone delivery's acceptance by the public opinion is divided. From one side are those who see the drone as a faster and more environmentally friendly alternative, and on the other side who are reluctant and fearful of the risks. Furthermore, drones' illegal activities have been the most diverse, such as terrorism, espionage, drug trafficking, violation of privacy, and airport disruption [72]. For example, since 2016, at least ten serious worldwide incidents involving drones in international airports resulted in serious losses [129].

Although legislation has improved in most countries [132], the authorities have faced enormous difficulties in reinforcing the law, i.e., to mitigate the illegal use of these machines [59].

Figure 1.1 – Miscellaneous of criminal usage of drones: terrorism, airplanes accidents, traffic of drugs and flight over nuclear plants.

1.1.1 Anti-drone solutions

In the military world, the combat against drones has been used for several decades [127], and consolidated techniques have been deployed since then. Although the Air Forces' anti-AUV systems may not be directly applied to the mini-drone neutralization, as exposed in [128], its extensive literature and know-how have been the starting point for much of the recent civil drone countermeasures. According to [57], most military techniques can be divided into the following classes: Jamming, Spoofing, and Mitigation.

While spoofing consists of inducing the drone a landing using false command signals, jamming, for its turn, disturbs the control with substantial signal interference, which can carry the drone to an emergency landing [57, 128]. However, these techniques can resent a relatively good efficiency [25]; they present a drawback of the short-range of actuation since their deployment is done conventionally by fixed ground stations. Furthermore, still after [25], interference techniques also present a relatively high cost, resulting in the demanded structure.

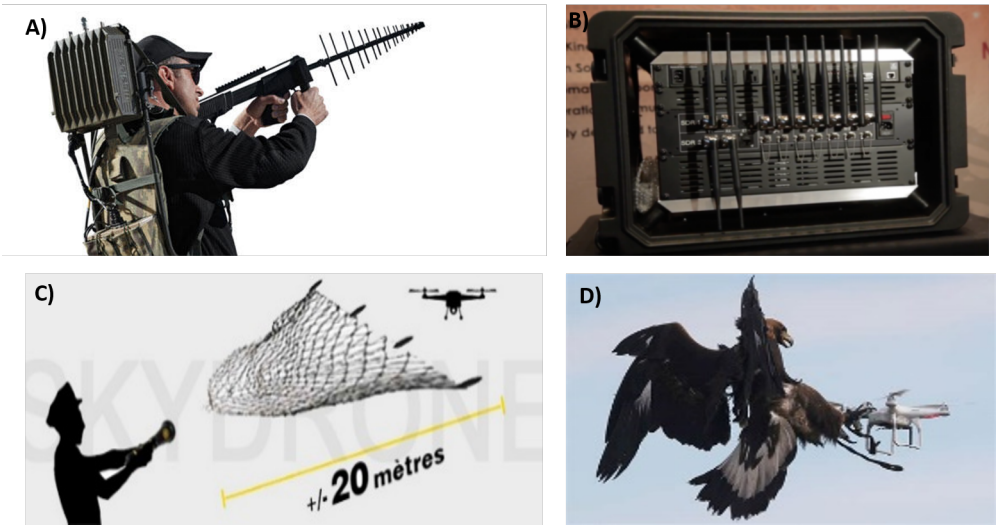


Figure 1.2 – Examples of anti-drone solutions. A) A policeman carrying a drone-jammer [136]. B) Equipment used for protocol manipulation (spoofing) [137]. C) Illustration of a net-launcher deployed by a policeman. D) Example of anti-UAV falconry.

The techniques of mitigation consist of destroying or intercept the invader physically [57, 128]. One emblematic example of mitigation was the "Anti-UAV Falconry" deployed by the Dutch National Police [135]. They studied the use of eagles and other prey birds to capture invasive drones. Evidently, this was a temporary exit, which carries numerous disadvantages, such as dependence on a bird's master, the animal's physical integrity, and its short-range actuation. Other examples of physical neutralization can be net launchers, water cannons, and projectile shooting [25, 129], which are generally deployed from the ground. Those techniques usually present the low cost of implementation but also have limited efficiency. Nevertheless, one particular case of mitigation, the laser weapons, such as the *Boeing's Compact Laser Weapon System* (CLWS), can offer excellent efficiency and a good range of actuation. But of course, all this in exchange for a very high installation cost.

The techniques presented so far do not have a viable cost or a good efficiency or operating range at once. Furthermore, all of these attempts are deficient in their potential collateral effects, i.e., the possibility of physical damage when the drone is neutralized. A drone landing forcibly or falling after an intercepting can cause harmful consequences in a civilian environment.

1.1.2 Anti-drone drone (ADD)

The anti-drone drone (ADD) is a recent technology that has stood out for its low cost, good efficiency, and potential for capturing the intruder and phase it out in a safe place. Furthermore, the ADD has an intrinsic advantage since it has the same "nature" of the threat, having essentially similar features. In this way, the countermeasure can evolve with similar speed to that of threat.

In recent literature, several works have been dedicated to ADD [3, 25, 27, 28, 62]. In [25], the author proposes a conceptual design of an ADD using a net launching system to neutralize the target. They considered the intruders' performance, the capturing system, and the missions' requirements in its design. In [3], [27], and [28], the authors dedicated to the trajectory planning for an ADD. In [27], the author proposed an improvement in the model predictive control scheme for allowing aggressive trajectories.

Moreover, in [3], the author chooses to compare missile guidance-based strategies with an optimal control planner. In [28], the author proposed an optimal controller based on a pursuit-evasion game formulation. They concentrated on a case where the target is partially observable, and they investigate the state-estimation as well.

While in the last examples, the authors considered the quadcopter as ADD aircraft, in [62], the author considered fixed-wing models. In this work, ADD behavior was modeled after a human-based model on aircraft combat pilots' experience. Besides, several ADD solutions are already commercially available. Companies, such as Theiss [130] and Hertz [131], for example, provide similar ADD solutions based on the net launch system. Both solutions are no-destructive and offer a low risk of damage in the civil application.

Figure 1.3 – Examples of anti-drone drone (ADD). A) An ADD equipped with a net-launcher. Similar solution has been commercialized by [130] and [131]. B) ADD with a fixed net deployed by the Tokyo police [154]. C) Kamikaze ADD developed the American startup Anduril [133].

Although an essential advantage of the ADD is the possibility of capturing the intruder, other kinds of interception have also been considered. For example, several works [27, 62] consider the kamikaze mode (or kinetic attack), where the pursuer attempts to crash into the target, disabling it. This attack can also be a backup mode in case of launch net failure. Another example is the American startup *Anduril*, which proposed an anti-drone system in which the kinetic attack is the means interception. Another possibility of neutralization is the use of interference guns on the drone. The Boreades project [134], proposes a counter UAV system where the ADD uses jamming guns to neutralize the intruder.

Until now, it has been evident the benefits of using the ADD in the counter-drone fight in civilian environments. However, since the ADD shares the same "nature" as its prey, i.e., they have virtually similar capabilities, the capture task could not be easy for the pursuer. The same technology that can allow autonomous pursuit can also favor autonomous evasion, making the one-vs-one pursuit a challenging task for the ADD. Nevertheless, to overcome the problem of a faster or stronger opponent, one recurrent solution observed in nature is cooperation, the use of a group instead of the single. In fact, group hunting increases the success rate of predators, as exposed in [45], and [33]. In a multi-agent system, the perception and the actuation area is significantly augmented, which can allow even the capture faster or more agile prey [52].

1.2 Collective motion

Collective behavior has been observed as an evolutionary advantage in many natural systems. Throughout the animal kingdom is evident how the group phenomena - as flocking, swarm, and pack - increases the rate of success of an individual living being significantly [44]. Even for the modern and corporate human being, teamwork is an appreciated characteristic of an individual. In short, humanity always new about the power of the collective, as the Aristotelian maxim said: “the whole is greater than the sum of its parts.”

1.2.1 Bio-inspiration

Understanding and modeling the complex behaviors observed in nature has been the primary motivation for many pioneers in multi-agent systems. Reynolds’ seminal work [49] aimed to graphically reproduce the behavior observed by flocks of birds and schools of fish. His studies observed that specific "laws" of behavior at the individual level, considering only local observation, could reproduce a verisimilar group behavior observed in natural systems. Namely, the three rules are: *separation* - a short-range repulsion to avoid crowded neighbors. *Alignment* - the tendency to align itself with its neighbors’ average heading. And *cohesion* - long-range attraction towards the center of the flock.

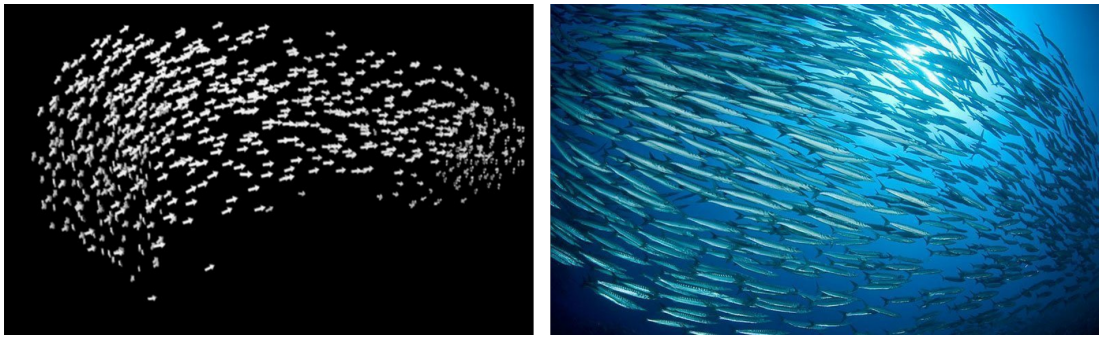


Figure 1.4 – One of the most classic flocking models [49] was created to graphically reproduce the behavior of collective animals, such as fish in a school. In the left figure, three-dimensional "boids" are represented [146]. On the right, a picture of a school of fish, from [147].

Universality of flocking Later, with the advancement of data acquisition techniques, observation studies found the flocking property in the most diverse groups of individuals, from bacteria, through insects to human groups on vessels, as evidenced in the Vicsek and Zafeiris' survey [50]. Individuals' tendency to collectively align their directions seems to indicate a certain universality of the collective motion. From Vicsek's seminal work [77], the physical community would explain this universal behavior as a result of the interaction of self-propelled particles. This collective movement, where individuals tend to align themselves fluidly, even in the absence of a leader or centralized coordination, is also called as *flocking*.

Models for flocking behavior are numerous in the literature, and they range from the simplest ones, such as primary Vicsek particle [77], passing through deterministic models, such as Cucker–Smale [138] and its numerous variation, and control-theoretical frame-work, such as in [58].

Flocking and hunting Although flocking behavior is present in practically all living beings moving in group [50], flocking models are not sufficient to describe all collective behavior types. Group hunting, for example, is a complex phenomenon that is not fundamentally described by flocking models. The description of lionesses' collectively hunting, carried out by Stander [45], indicates the lionesses' previous tendency to disperse and then ambush the prey. Disper-

sion behavior is contrary to the aggregation trend described in most flocking models. This difference is evident in the work of Janosov [52]; the authors proposed the addition of a dispersion term into the conventional flocking model [53, 54] to obtain a more realistic hunting modelization.

Furthermore, Reynold's alignment rule, with its correspondent in the field of statistical physics, the viscosity term, acts as a force against a reactive hunting movement. The alignment makes the direction of a stalker tend towards its neighbors' average direction, which may be contradictory with the tendency of a hunter who orientates above all with respect to his prey. Some recent multi-agents approaches have tried to model group hunting using principles of collective motion. Later in this chapter, we will make a brief review of the emerging field of group chase-evasion.

1.2.2 Collective robots

It is possible to highlight several similarities between group animals and collective robots, such as limited sensing and communication and decentralized decision-make. These similarities explain why flocking models have been widely integrated into robots, as proposed in [53, 74, 139] and in many other works [140].

However, as expressed in Viragh [53], real-life conditions, such as delays, communication failures, and uncertainty, can cause instability among agents. Consecutively, the design of multi-robots must go beyond the theoretical flocking model, and the physical characteristics of the robot and its sensors must be taken into account. Therefore, multi-robots can be seen as an extension of the classic models of group movement models, as described in the previous section, but taking into account the robots' dynamics and observation.

A vast literature collective robot (or swarm of robots) has been formed in the last two decades, and several efforts have been applied to organize and classify them, such as in [140, 141]. For the sake of clarity, we consider henceforth a classification similar to one proposed in [140], which classifies the systems of multi robots based on their emergent behavior. For example, as a collective behavior,

we can highlight the following group tasks: reaching consensus, coordinated motion (flocking), or pattern formation. Next, we will review two of the most popular behaviors applied to aerial robots and their application to group chase.

Pattern Formation

In the field of aerial robotics, impressive results have been achieved in fixed and dynamic pattern formation, such as described in Figure 1.5. Autonomous formation control has been extensively addressed in the control and robotics literature. This approach aims to create geometrical patterns with the agents by controlling their relative positioning.

The main control structures used to achieve pattern formation are namely: Leader-follower [69, 70, 71], Virtual Structure [80, 142] and Behavioral-Based [58, 83]. Although there are huge differences between the architectures, in this work, we will stick only to their final results, which are the formation of patterns themselves. Here, we are interested in whether the formation of patterns is interesting or not for group persecution.

Figure 1.5 – Examples of real-time applications of pattern formation with quadcopters. In the left picture, one of the earliest works with dynamic pattern formation [142]. In the right picture, picture of the Intel Drone Light show at the Winter Olympics of 2018 [144].

Several attempts have been made to apply pattern-formation techniques to the group chase problem. In one of the earliest works [30], the author proposed a feed-back control to regulate a non-holonomic pursuer's position in a "stalking" formation around the target. Each agent's behavior was defined by a "formation vector," and the resulting pattern was a half-circle around the target. In [32], the

author proposed a bio-inspired wolf-pack's model for a multi-robot application. This proposition involves a hierarchical system, and a finite state machine determines the robot's motion; nevertheless, the system relies on a fixed formation and fixed roles.

Moreover, other commons approaches, such as in [46, 47], the encirclement is achieved by each pursuer tracking a fixed point around the "virtual target." In this case, the problem becomes a collision free trajectory generation problem, which has been solved by genetic algorithms [46] or virtual ranges [47].

Although the previous approaches could provide an encircling formation around the target, they did not consider the target dynamics. In [31, 48], the target movement is considered, and the pursuit is made toward a predicted position target. Nevertheless, both techniques still rely on virtual points around the predicted position of the target. Once more, the problem becomes the generation of trajectories toward a virtual position. In a most recent work [84], the author proposed a motion controller for a group of drones towing a capture net to intercept an intruder UAV. Here, since the net physically links the agents, the maintenance of the fixed pattern is needed.

In short, the use of pattern formation in the pursuit, although it can bring some advantages to groups, such as increased perception, robust and acted area, is still a very rudimentary technique of pursuit, which reduces the flexibility of the group in a more challenging pursuit. The formation can be seen as a single body, and the most advanced techniques of ambush and stalking require more than the maintenance of the formation. Furthermore, most of the above-cited techniques require knowledge of the target and neighbors position in the inertial frame, which requires communication between all the pursuit actors.

Flocking of robots

Flocking is another prevalent collective behavior for aerial robots. Here, we consider multi-robots techniques which are based or similar to the "flocking" models seen previously, such as Reynolds [49] and Vicsek [77] models. We suppose that the agents do not follow a fixed path or relative positioning in a pattern formation; instead, they are independent members of a cohesive group, sensing and acting locally and based on simple rules.

Figure 1.6 – Examples of real-time applications of flocking with quadcopters done by Vásárhelyi et al [148]. Thirty drones are fly in an virtually confined space.

As pointed out by [140], flocking behaviors with multi-robots are generally designed based on virtual physics, where virtual forces of attraction and repulsion ensure the distance between agents and the alignment ensures the coherence of the movement.

One of the first successful attempts to reproduce Reynolds rules with aerial robots is given in [74]. In this work, ten fixed wings UAVs were deployed, and they focused on analyzing the effects of communication range and maximum turning rate in the emergent formation. Other impressive examples of UAV's flock are given in [53, 54, 80, 148]. The authors deployed a fleet of over large numbers of quadcopter flights in an entirely autonomous and decentralized manner in those works. In those works, the flocking algorithm is based on the in the Vicsek model [77]. Pictures of the real-time experiment can be seen in Figure 1.6.

For the best of our knowledge, no flocking based strategies applied to group

pursuit have been implemented in real-time robots. Nevertheless, several approaches have been made in simulations, such as [33, 35, 51, 52]. Such approaches will be better described in subsection *Pursuit-evasion*.

1.3 Pursuit evasion

Similar to flocking, pursuit-evasion is also a very popular phenomenon in nature, which can be easily observed in several biological activities, such as foraging, searching for partners, and disputing territories [37]. As we will see below, persecution has a long mathematical tradition; however, the recent computational advances have allowed a new approach to the problem, allowing more complex ones, with large numbers of persecutors and evaders.

1.3.1 Single Pursuit-evasion

Pursuit's curbs Although the problem stated dates from farther away, the first successful attempt to describe the trajectories of pursuit is attributed to Pierre Bourger, in *Les courbes de Poursuite* (1732) [14, 37, 106]. Although the problem can be stated simply, closed-form analytical solutions are hard to obtain, even using simplistic assumptions such as constant velocities and linear trajectories. For example, a generalized extension of the closed-form solution of Bourger was only obtained late in 1991, as exposed in [106].

Differential game This approach formulates the pursuit-evasion as a two players game. The players (decision-makers) must maximize or minimize an objective function, which often is the duration of the episodes. Such formulations have been extensively studied in modern robotics problems. In [64], the authors made an interesting survey highlighting game-based approaches applied to robotics; consequently, the main classes of techniques were Pursuit-Evasion Games and Probabilistic Search. The authors remarked a tendency to use the second one in adversarial games, where the optimization expects to maximize the evader detection probability. In [65], the authors implemented this technique in a heterogeneous team with ground and aerial vehicles. They demonstrated

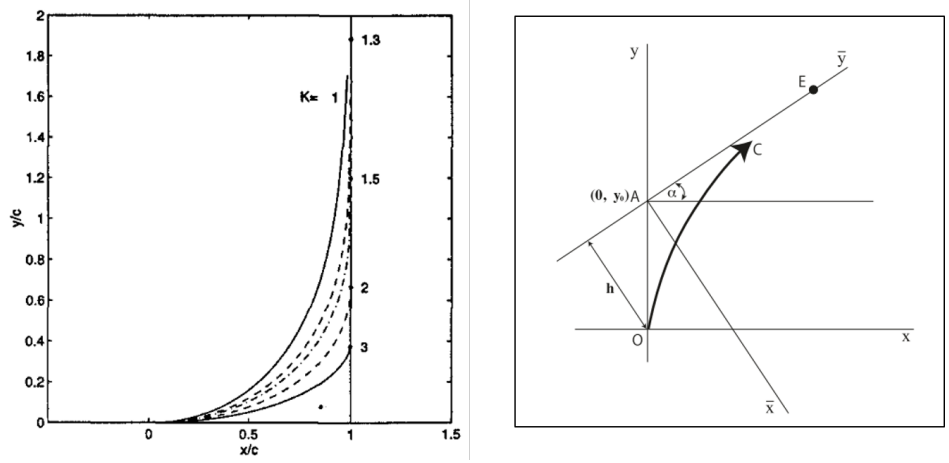


Figure 1.7 – Curbs of pursuit. Left: the classic Bourger's solution for different velocity's ratio between target and pursuer, extracted from [36]. Right: Example of the generalized closed-form solution for a 2D pursuit, exposed in [106]

empirically that their method could locate a target but with no guaranteed capture.

Others work using pursuit-evasion games in complex environments are available in [66], [67], and [68]. In those works, we can find the following configuration consecutively: convex polygons arenas, with obstacles and no-fling zones.

The multi-pursuer single-evader game is also a scenario extensively studied in the literature. In [42], the authors used a differential game formulation for multi pursuers chasing a single evader. In their proposition, each agent minimizes the evader's action area, characterized by the Voronoi cells, similarly, in [43], a cost-function is proposed and computed based on the relative angle and distance between agents and targets, seeking pursuit and avoid a collision.

Although game-based methodologies guarantee the optimality input for pursuit, the large amount of constraint and precise knowledge about the whole system makes it, for instance, impracticable for real robots application. Moreover, it becomes more challenging to define an appropriate objective function with increasing problem complexity, such as a larger number of agents or motion constraints. Finally, as exposed in [52], the apparatus of the differential

game formulation requires a lot o "unrealistic and biologically questionable assumptions."

1.3.2 Group pursuit

The emerging "group chase escape" analysis has raised a recent early interest in the physics and biology communities. This field aims to describe prey-predators' complex behavior considering agent-based models, such as in [34, 51, 52]. Moreover, as pointed by [85], this recent and opened field inherits from two consolidated literature: *pursuit-evasion* and *collective motion*.

Firstly, it can be seen as an extension of the single pursuit-evasion problem, taking advantage of an extensive literature involving: analytical solutions of pursuit, differential games, and missiles guidance studies. Secondly, it can be considered an extension of the collective motion, taking advantage of the extensive literature in self-propelled particles, such as [77], which has been a powerful tool model complex systems, such as bacteria, insect, and animals [50].

The first proposal to bring together chase-evasion and multi-particle systems is possibly from Kamikura et al. [153]. In this work, the authors established a multi-agent pursuit-evasion problem running in a grid world with periodic boundary conditions. The pursuers, which have no interactions between themselves, had to capture the nearest evaders, while the evader must escape from the closest pursuer. Despite the simplicity of the model, interesting behaviors emerged, raising the scientific community's interest.

Although those earlier works, such as [51, 152, 153], allowed the first analysis of emergent behavior, such as segregation and encircling, they had elementary assumptions, which kept it far from real-time applications. Nevertheless, since then, approaches with more realistic simulations, such as continuous and boundary world representation, were proposed [33, 34, 52], and the group-chase started to become closer to robotic applications.

In [33], the author implemented an agent-based approach, based on simple rules, to mimic a wolf-pack hunting strategy. The authors pointed out that there was no need for communication to mimic encircling behavior. They showed

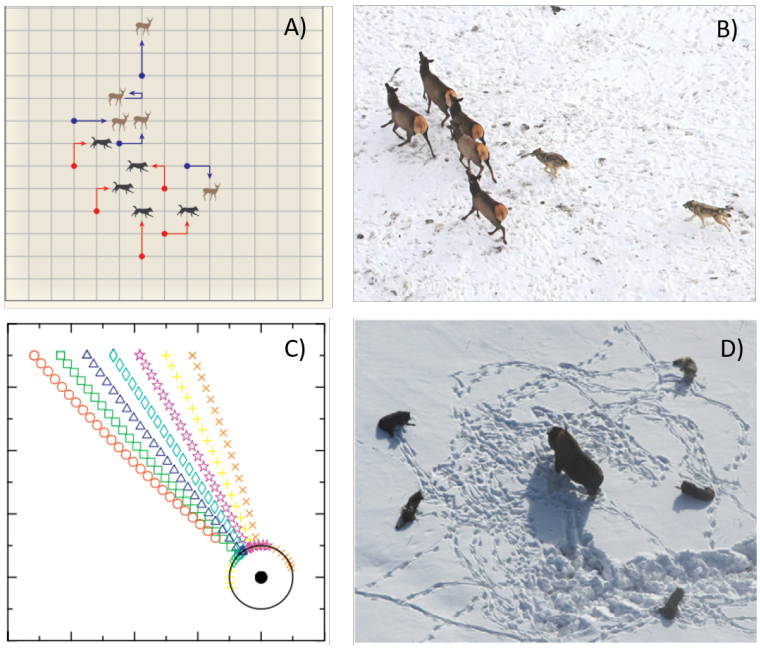


Figure 1.8 – Group-chase algorithms and its bio-inspiration. In *A* a screenshot from [152], it is illustrating the initial positions of pursuers (wolves) and evaders (deers) using the algorithm proposed by [153]. *B* illustrates the motivation of the previous: modeling the group chase evasion natural phenomenon. In *C* a screenshot from [33], where the authors proposed to mimic a wolf-pack's behavior, as illustrated in *D*.

that simple rules of attraction toward the target and regulating distance between pursuer were enough to reproduce this performance. However, their algorithm does not describe the complexity of a cooperated pursuit of faster prey, which requires more sophisticated techniques to stalk and ambush the prey. In another agent-based approach, [34] constructed the model based on the Vicsek particle, and [35] modeled a similar multi-agent hunting system but in a three-dimensional environment.

Recent work in escape from group chase [52] extended the previous particle-based model [34] to more realistic simulations, where delays, inertial effects, and three-dimensional work-space were considered. In this work, the pursuer behavior is calculated by the sum of the virtual interactions: chase (attraction to the prey), inter-agent repulsion, alignment, and short-range collision-avoidance.

Furthermore, they considered the prediction of prey's future position to improve the verisimilarly of the hunting.

Although impressive behaviors can be exhibited in the simulations, their model considered many tunable parameters, which increased the difficulty of tunning and required optimization. Furthermore, its applicability in the real world is limited due to its assumptions in the agents' observation; it relies on the knowledge of position and velocities of pursuers and targets expressed in the global frame. Although these assumptions can be partially afforded by considering communications between pursuers, the evader is not cooperative in real-world applications; consecutively, its global position and velocity usually are not available for the pursuer.

1.4 Guidance Laws (GL)

Guidance is a crucial part of any autonomous vehicle system. As once defined by Shneydor [36], guidance can be said: "*the process for guiding the path of an object towards a given point, which in general may be moving*". Considering a complex robotic complex system, which involves several layers in the motion control, the guidance layer can be considered high-level control, which indicates the direction to be taken.

Here, "guidance laws" refer to feedback algorithms in which a particular geometrical rule is implemented. The geometrical rule is the essence of GL and expresses how the pursuer must behave in geometrical terms. For example: "head the vehicle towards the target's current position," which is the rule behind the Pure Pursuit (PP) law. Since GL is an essential concept in this thesis's propositions, we will give more detail in the next subsections.

1.4.1 GL in robotics

As pointed out by [75], possibly the richest literature in guidance can be attributed to the missiles' community. This fact is not strange considering that

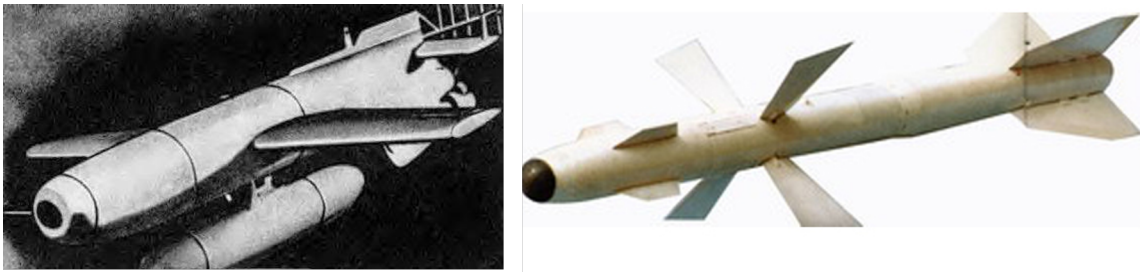


Figure 1.9 – Guidance laws have been widely applied in guided missiles. Left: the German TV guided bomb Hs-293D, considered the first PP-bomb. According to [36], an operator transmits commands to the bomb according to its location on the screen. Right: Guided Missiles R-27T1, an example of modern air-to-air missiles, which implements PN based guidance law [151]

missiles were one of the first autonomous vehicles and, in a way, even the predecessor of the current UAV [150].

GL for ground robots Considering ground robot applications, GL's have been extensively explored, see [38, 39, 40]. [38] investigated GLs to guide a single robot towards an evader in an environment with obstacles. They applied two classical guidance approaches: Velocity Pursuit Guidance (VPG) and the Deviated Pure Pursuit (DPP). In [39], the authors implemented a feedback controller based on Proportional Navigation Guidance (PNG). Similarly, in [40], the authors applied a sliding-mode based control to keep a constant line-of-sight, which is the geometrical rule for the PNG.

GL for UAVs Classic guidance laws application stills pique interests on the scientific community, and it has been widely used in aerial robotics, such as in [1, 3, 28, 41]. In [3], the author applied PP and PNG guidance algorithms into quadcopters for target interception. They compared GLs against optimal control trajectory planners, analyzing the time and energy required to achieve the task. Nevertheless, results are only presented in simulations. In [28], the author analyses three missiles guidance laws applied to interception trajectory in the case of a partially-observable target. They formulate a pursuit-evasion game to determine the guarantees of capture under evasive maneuvers, determining an optimal controller.

In [41], the authors used the PNG scheme in a drone landing on a moving platform. Its final solution mixed PNG with PD - proportional derivative - algorithms, and it showed effectively and carried out less oscillation and small errors compared to the traditional tracking PD. Moreover, the PNG has also been used as guidance law in an autonomous drone race in [1]. In that case, the authors adapted the PNG algorithm to handle the quadrotor's decoupled dynamics between roll and yaw to follow visually desired trajectories in a drone race.

1.4.2 Relative engagement

The analysis of the GL is based on the relative kinematic between pursuer and target. This kinematics equation can be easily deduced from the geometrical engagement of pursuer-target, as represented in Figure 1.10.

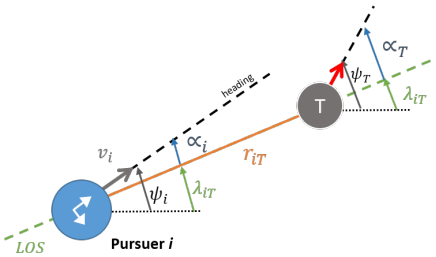


Figure 1.10 – Geometrical engagement between pursuer and target.

From Figure 1.10, the straight line, also called Line-of-Sight (LOS), connecting them is denoted by r_{iT}. The bearing angle, formed by the LOS and the inertial axis-x, is denoted here by λ_{iT}. The vector v_i is the straightforward velocity in the body frame, and ψ_i represents the heading (yaw) of the pursuer i.

As explicated in [17, 36], from this engagement a relative kinematic model can be obtained. For that, lets consider the relative velocity between, $\dot{r}_{iT} = \dot{P}_T - \dot{P}_i$. The relative velocity can be decomposed along ($v_{\parallel}$) and perpendicular ($v_{\perp}$) to the LOS :

$$\begin{aligned} v_{\parallel} &= \dot{r}_{iT} = v_T * \cos(\alpha_T) - v_i * \cos(\alpha_i), \\ v_{\perp} &= r_{iT} * \dot{\lambda}_{iT} = v_T * \sin(\alpha_T) - v_i * \sin(\alpha_i). \end{aligned} \tag{1.1}$$

where $\alpha_T = \psi_i - \lambda_{iT}$ and $\alpha_i = \psi_T - \lambda_{iT}$. In other words, with this equation we have represented the velocities of target seem by the pursuer along the the LOS.

These equations will be useful to determine conditions of convergence, or capture, for the proposed pursuit strategies.

1.4.3 Pure Pursuit (PP)

In the most intuitive guidance law, the Pure Pursuit law, the pursuer aims to align its heading (or velocity vector) towards the present location of the target ($\psi_i \to \lambda_{iT}$). Therefore, assuming this perfect alignment ($\psi_i = \lambda_{iT}$) the relative kinematics equations 1.1 become:

$$\begin{aligned} \dot{r}_{iT} &= v_T * \cos\alpha_T - v_i, \\ r_{iT} * \dot{\lambda}_{iT} &= v_T * \sin\alpha_T. \end{aligned} \tag{1.2}$$

Note that, a sufficient condition for the capture is that $v_i > v_p$, once this condition assures $\dot{r}_{iT} < 0$, for any values of α_T. This statement evidences the limitation of the use of a single pursuer against a faster target. However, this problem will be better discussed in Chapter 5.

Finally, considering the kinematic model of an agent (3.16), an intuitive law to apply the PP ($\psi_i \to \lambda_{iT}$) is:

$$f_{\psi_i} = -K_{pp} * (\psi_i - \lambda_{iT}) \tag{1.3}$$

where K_p defines a positive constant gain.

1.4.4 Pure Deviated Pursuit - DPP:

As stated by Shneydor1998, the DPP can be seen as a variation of the classical PP, it can be a result of manufacturing default or can be intentionally designed in order to point the missile in position ahead of the target, causing a "lead pursuit." However, if the offset drives the motile towards a position behind the target, it causes a "lag pursuit."

Its basic principle relies in driving the heading error not to zero by to a constant ($\psi_i - \lambda_{iT} \rightarrow \alpha_0$). Once considering this principle, the relative kinematics become:

$$\begin{aligned} \dot{r}_{iT} &= v_T * \cos(\alpha_T) - v_i * \cos(\alpha_0), \\ r_{iT} * \dot{\lambda}_{iT} &= v_T * \sin(\alpha_T) - v_i * \sin(\alpha_0). \end{aligned} \tag{1.4}$$

From the above equation, we can established one necessary condition about the offset angle: $|\alpha_0| < \pi/2$. Nevertheless, under this strategy the capture cannot be assured for all $v_i > v_p$.

The further interest in applying this technique is not to choose a α_0 that optimizes the pursuer displacement. Instead, the idea is that each agent assumes different offsets, consecutively, having different degrees of lead, and lag pursuit, spreading the pursuers around the target. In Chapter 2, one group pursuit strategy will be established based on this technique.

Finally, as exposed in [36], an intuitive law to apply the DPP ($\psi_i \rightarrow \lambda_{iT}$) is:

$$f_{\psi_i} = -K_{dpp} * (\psi_i - \lambda_{iT} - \alpha_0) \tag{1.5}$$

where K_{dpp} defines a positive constant gain.

1.4.5 Proportional Navigation Guidance (PNG)

The Proportional Navigation is maybe the most popular guidance law for missiles. Its basic principle is called Parallel Navigation or constant bearing rule. It consists of keeping a constant bearing angle (λ_{iT}) with respect to the target, and since the LOS rate ($\dot{r}_{it}$) is decreasing, both elements are in the collision route.

The same principle was observed in animals hunt, as the reputed faster predator, Falcon peregrine, cited in [22], and also in insects, as described in [19], where miniature brains predators execute fast maneuvers to catch its prey.

As exposed by Shneydor in [36], for the planar case, the parallel navigation rule can be stated as:

$$\begin{aligned} \dot{\lambda}_{iT}(t) &= 0, \\ \dot{r}_{iT}(t) &< 0. \end{aligned} \tag{1.6}$$

The above equations are also known as "straight-line collision course conditions". Applying this principle (1.6), to the relative kinetic model (1.1), the following relation can be obtained:

$$\begin{aligned} v_T * \sin(\alpha_T) &= v_i * \sin(\alpha_i), \\ v_T * \cos(\alpha_T) &< v_i * \cos(\alpha_i). \end{aligned} \tag{1.7}$$

If the above conditions are satisfied and the velocities remain constant, the straight-line collision course conditions prevail, and both bodies will converge.

We can see from (1.7) that, contrary to the Pure Pursuit (1.3), solutions can exist even for $v_T > v_i$. In other words, depending on the initial conditions of the engaged, a faster target can still be in a collision course with a slower pursuer. This property will be explored in Chapter 2, where the faster evader case will be studied.

Finally, a logical choice for a controller to implement the PNG can be defined as:

$$f_{\psi_i} = N * \dot{\lambda}_{iT} \tag{1.8}$$

where N is a positive constant called the navigation gain.

1.5 Conclusion

Nowadays, there is a consistent threat by the unlawful use of civilian drones. The classic counter-UAV solutions have proven to be insufficient, and there is a growing trend in the use of anti-drone drones (ADD). However, in a futuristic

but next scenario, with autonomous intruder drones, the use of a single ADD may be insufficient, as the intruder may have similar capabilities to the ADD. In this case, it would be relevant to use a group ADD to cooperate and ambush the intruder, similarly as predators in nature do.

However, coordinating robots for a chase is not an easy task. Although group robots are becoming popular and extensive literature exists, the existing multi-robots techniques are not appropriate for group pursuit problem. On the contrary, most of the existing techniques are in charge of pattern formations or flocking navigation, which is very different from a cooperative hunt.

On the other hand, the study of pursuit-evasion has been a topic widely addressed and implemented in the military community. Although the existing single-pursuit analysis can serve as insights for group-pursuit, the large degree of freedom of this second makes the problem practically unfeasible to be solved analytically with realistic conditions. In this context, *group chase-evasion* arises, which uses multi-particle simulation techniques to analyze complex collective motions.

Finally, our work comes in the context of group chase-evasion strategies and aims to contribute to this field with an approach more adapted to aerial robots' real applications.

"But ask the animals, and they will teach you, or the birds in the sky, and they will tell you;

Job

Chapter 2

Pursuit in the horizontal plan

In Chapter 2, we study the scenario where a group of pursuer and a single evader are traveling in a bounded 2-dimensional workspace. To solve the capture problem, we propose three distinct behavioral-based strategies for the pursuer team. The first one is based on the *Deviated Pure Pursuit* guidance, the DPP, and the different offset values give, for each pursuer, different behavior in the stalking. The second strategy is an improvement of the previous one, where an *Parallel Navigation* term is added to the DPP. This allows the pursuer to take more efficient paths toward the target and still deviate from each other, avoiding collisions. Finally, we propose a Deep Reinforcement Learning framework to obtain close-to-optimal policies for the pursuers.

Contents

2.1 Problem statement

Let us consider multi-agents pursuit-evasion problem involving N pursuers and a single evader moving in a horizontal plane. The pursuit takes place inside a bounded work-space W in $\mathbb{R}^2$. The final goal for the pursuer team is $r_{iT} < R_c$, i.e. that at least one of the pursuers has a relative distance (r_{iT}) towards the target inferior than a threshold R_c. The evader paths are described by $P_T(t) = [x_T(t), y_T(t)]$, where (x_T, y_T) is its cartesian coordinates in W. Similarly, the pursuers path are described as $P_i(t) = [x_i(t), y_i(t)]$, where $i \epsilon 1, 2, ...N$.

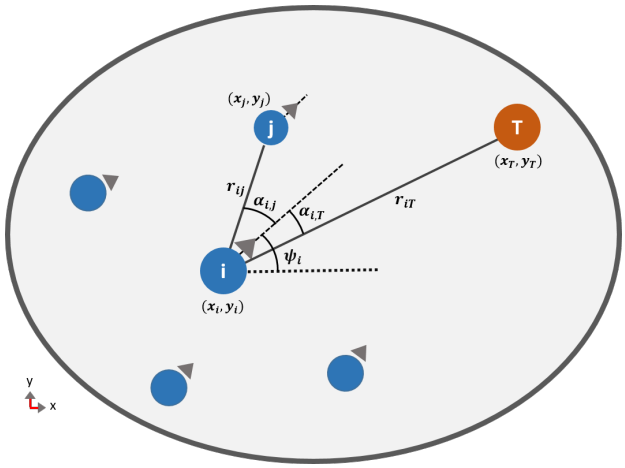


Figure 2.1 – Group pursuit in a bounded work-place. The pursuer (blue circles) attemps to capture the evader (red circle). Furthermore, it is in evidence the relative engagement between the pursuer i and one neighbors j and the target T.

Pursuer and target are moving in the work-space with a linear velocity (v_i) and heading (ψ_i). The motion model can be given by:

$$\begin{aligned} \dot{x}_i &= v_i \cos \psi_i, \\ \dot{y}_i &= v_i \sin \psi_i, \\ v_i &= f_{vi}, \\ \dot{\psi}_i &= f_{\psi i} \end{aligned} \tag{2.1}$$

which $\boldsymbol{f}_i = [f_{vi}, f_{\psi i}]^T$ the control vector of the particle. The turn-rate and velocity control are saturated, being $f_{vi} < v_{max}$ and $f_{vi} > 0$ and $|f_{\psi i}| < \omega_{max}$.

Note that the non-holonomic equations (2.1) describe only the agents used in simulation; the quad-copter dynamics under the pursuit will be discussed further chapters. Nevertheless, it is worthy to lies some advantages of choosing this kinematic model for pursuer agents:

- It is a model compatible for a large number of vehicles, such as car-like, airplanes, etc.
- Most of the works dealing with the pursuit problem rely on this hypothesis, see Chapter 1.
- Even for holonomic robots, which is the case of the quadcopter, these constraints can be useful to conciliate the perceptions constraints (sense-and-go), see in Chapter 4.

The assumptions of v_{max} and ω_{max} will vary along with this thesis, and it will be evidenced at the beginning of each experiment. In a similar way, the number of pursuers N, the initial positions, and the shape of W can vary along with this document.

2.1.1 Perception

Limited sensing is a common feature between multi-agents solutions [34, 49, 50]. Nevertheless, in this work, we decided to consider the perception of the environment (neighbors, target, obstacle, and boarders) given in relative and polar polar coordinates, as illustrated in Figure 2.1. This assumption is justified mainly for two reasons:

- Considering the robotics application, relative and polar coordinates are an expected output from commons embedded sensors, such as lidar, sonar, and camera.
- Most of the pursuit-evasion analysis are done based on this coordinate system; thereby, we could take advantage of its extensive literature.

Many works on drone's pursuit, such as [3, 27], or on navigation in a complex environment, such as [1], rely mainly on the frontal camera for the environment sensing.

2.1.2 Target behavior

The ultimate advantage of using multi-agents in pursuit is the possibility of a cooperative ambushing for intercepting a more agile target. The term "agile" can mean having superior velocity or having more degrees of freedom. Along with this thesis, both of these hypotheses will be assumed.

In the following, the intruder will be up to twice as fast as the pursuers. Furthermore, the intruder will not be constrained to non-holonomic equations 2.1, but by a simple particle model, which gives him locomotion advantages. This configuration is a very known problem in differential games, and it is called "the homicidal chauffeur," [79].

Two strategies are assumed in most of the paper regarding the target behavior: predefined trajectories and reactive evader. First, the trajectories can be off-line defined; for example, by collecting real-data flights, or functions or algorithms can give it as circular or square trajectories. In the second, the reactive behavior is defined by a repulsion model. Each pursuer and the arena border exerts a repulsive force towards the evader. The forces decrease proportionally to the distance squared. The instant target's velocity is calculated as follow :

$$\boldsymbol{v_i} = \sum_j \frac{P_j - P_T}{r_{jT}^2} + \frac{r_{arena} \times (\sin(\gamma), cos(\gamma)) - \vec{pos}}{d_{wall}^2} \tag{2.2}$$

where γ is the direction of the agent to the closest point on the wall.

2.2 Group Deviated Pursuit

This section presents our first attempts to model the collective hunting phenomenon using Navigation Guidance Laws (NGL). This simplistic strategy bases on the Deviated Pure Pursuit law (DPP), and it is inspired by the behavior of a group of lionesses hunting.

The content of this section is currently under the second revision for a journal publication. The paper is entitled "**Local interaction and navigation guidance for hunters drones**." C. de Souza Jr, P. Castillo, B. Vidolov.

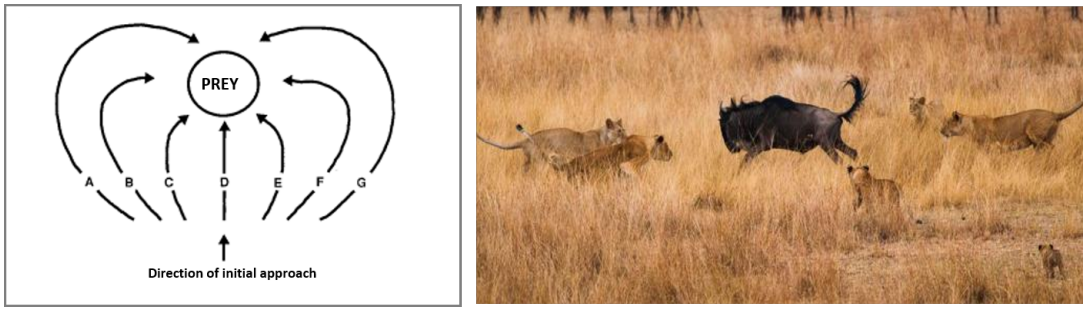


Figure 2.2 – Lionesses hunting formation. The left figure, extracted from [45], illustrates the distribution of a group of lionesses towards the prey. The right figure shows the final moment of a lionesses group attack.

2.2.1 Opening remarks

Imitation of nature has always been an excellent starting point for many robotics technologies. With that in mind, we decided to seek collective behaviors in nature that could be interesting for our problem, multi-agent pursuit. Therefore, in lions hunting, we find an interesting analogy with robotics' application; since lionesses are usually individual hunters but can cooperate in the presence of same pride individuals. This ability can be a desirable behavior for a fleet of stalking robots, in the sense that they are not dependent on each other, neither in the formation itself; however, they can cooperate for the endeavor in the presence of neighbors.

[45] analyzed the cooperative hunting of groups of lionesses (*Panthera Leo*) in the semi-arid plains in Namibia. The author observed that some individuals tend to orient themselves directly towards the target, while others tend to take "indirect routes" or even encircling the prey. He classified the hunting roles according to its positioning in the formation: *wings (right and left) and centers*, alluding to football players' roles. The "wings" ambush (taking indirect routes) and drive the prey towards the center. The resulting formation is described in Figure 2.2. Moreover, in the lioness' case, the roles are most likely assigned to the same individual.

In our work, the term "role" will be employed similarly to [45], which refers to the "trajectory" adopted by the pursuer towards the target. Later on, we will see that, by using the Deviated Pure Pursuit (DPP), the trajectory's shape can be modified by changing an offset value. However, other than the observed by [45], in our approach, the roles are not assigned to one individual, rather they are given based on the current engagement with the prey and its neighbors, as illustrated in Figure 2.3.

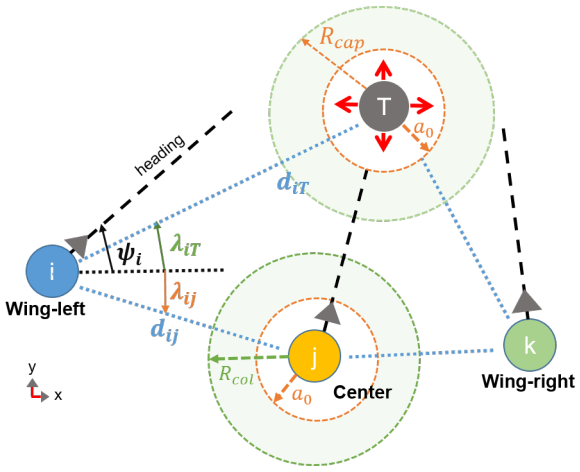


Figure 2.3 – Engagement between three pursuer and a target.

Furthermore, in this approach, unlike most other multi-agents techniques, the agents have no constraints about fleet formation or cohesion. Instead, each agent goes toward the target and adapt its route in the presence of neighbors.

The absence of repulsive/attractive forces in the formation avoids a common

oscillation problem (spring effect), accentuated in real-world applications, where delays, loss of data, or long sample times are present [53].

Finally, our strategy relies, *a priori*, on bearing-only information about the target and neighbors, which can be easily obtained from ordinary onboard sensors, such as a camera or lidar. Furthermore, the assumption of bearing-angle information solves the recurrent problem of observing the neighbors' position/velocity, which often requires explicit communication premises.

2.2.2 Group Deviated Pursuit (GDP) algorithm

In order to introduce our strategy, let us consider a conventional form of the DPP, exposed previously in (3.9). By considering different values offset angle (for $\alpha \in \mathbb{R} | -\frac{\pi}{2} < \alpha < \frac{\pi}{2}$), different trajectories can be observed, such as in Figure 2.4. Notice also that the Pure Pursuit (PP) can be described as an especial case of DPP with $\alpha = 0$.

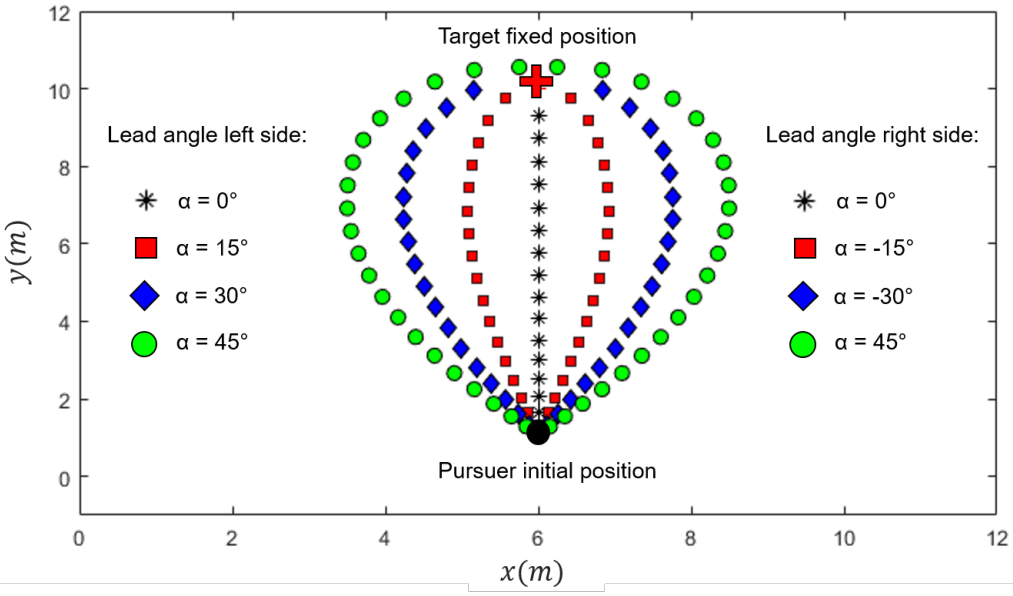


Figure 2.4 – Paths of deviated pursuit according to different offsets of the angle α. The agents begin the trajectory in the bottom part and keep a constant velocity until they get the target.

As stated previously, the "roles" in this strategy are related to the pursuer's trajectory shape. Therefore, the idea is that each agent can assume different references of α, causing consecutively different trajectories and increasing the possibility of capturing the intruder. Thus, let us consider a variable offset angle of the agent i (α_i) given by:

$$\alpha_i = \alpha_0 * \sum_{j!=i}^{N} \delta_{i,j}(\xi_i, \xi_j)$$

where N is the number of neighbors of the i-agent, and α_0 is a constant offset step. The δ_{ij} is a function indicating the positioning of agent j in the Pursuit frame of the agent i (P_i). Thus, δ_{ij} can be defined as:

$$\delta_{i,j} = \frac{y_j^{P_i}}{|y_j^{P_i}|} \tag{2.3}$$

where $\delta_{i,j}$ is calculated for each neighbor, and it basically indicates in which side the neighbors j is in relation to the λ_{iT}: left side $\delta_{i,j} = 1$ and right side $\delta_{i,j} = -1$. In equation (2.3), $y_j^{P_i}$ represents the lateral position of the agent j in P_i. The origin of P_i coincides with body frame of i, nevertheless it is rotated in $(\lambda_{iT} - \psi_i)$, in such way that its x-axis is always pointing to the target. To obtain the coordinates, $\xi_j^{P_i} = [x_j^{P_i}\ y_j^{P_i}\ z_j^{P_i}]^T$, of the agents neighbors j in the P_i frame, the following relation is used

$$\xi_j^{P_i} = T_I^{P_i} * \xi_j \tag{2.4}$$

where, $T_I^{P_i}$ is a matrix transformation representing inertial position ξ_j in the pursuit plane P_i. Finally by substituting (2.3) in (3.9), the final guidance law for a multi-agent pursuit can be obtained:

$$f_{\psi_i} = -K_{dpp} * \left(\psi_i - \lambda_{iT} - \alpha_0 * \sum_{j!=i}^{N} \delta_{i,j}(\xi_i, \xi_j) \right) \tag{2.5}$$

2.2.3 Numerical Validation

In order to analyze the performance of the algorithm (2.5), we considered the framework proposed in the section 2.1, where the agents modeled as a particle with constant velocity and moving in a plane. The guidance law (2.5) was applied, and the following three scenarios were considered:

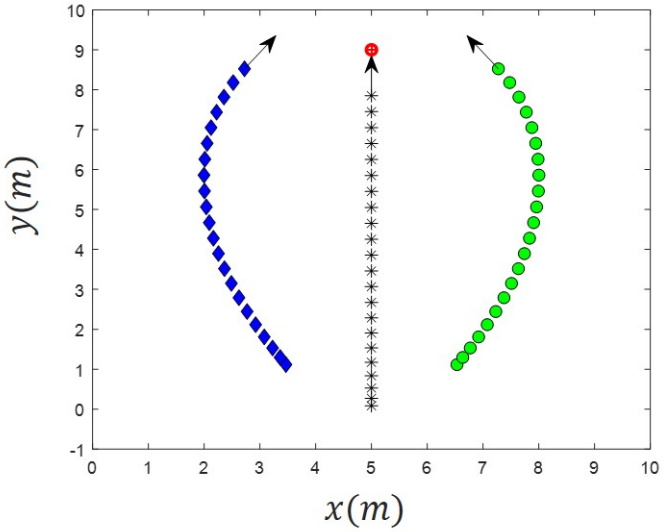


Figure 2.5 – Trajectories generated by three agents and a fixed target. The pursuers start at the bottom of the square, and they pursued the target with constant velocity. The deviation offset (α_0) for this pursuit is 30 degrees.

Fixed target: The first simulation is done for a simple scenario where the intruder drone stays in a fixed position. It is proposed to illustrate the formation pattern in a simple case. Notice from Figure 3.7 the patterns of "wings" and "center" can be determined with three pursuers

Escape trajectories: Several simulations were carried out with a different number of pursuers (1, 2, 3 and 4), see Figure 2.6. For these simulations, the intruder velocity was set twice as fast as the pursuers. Besides, in this scenario, we propose reactive escape behavior for the evader. We implemented the repulsive behavior, as described in subsection 2.1.2.

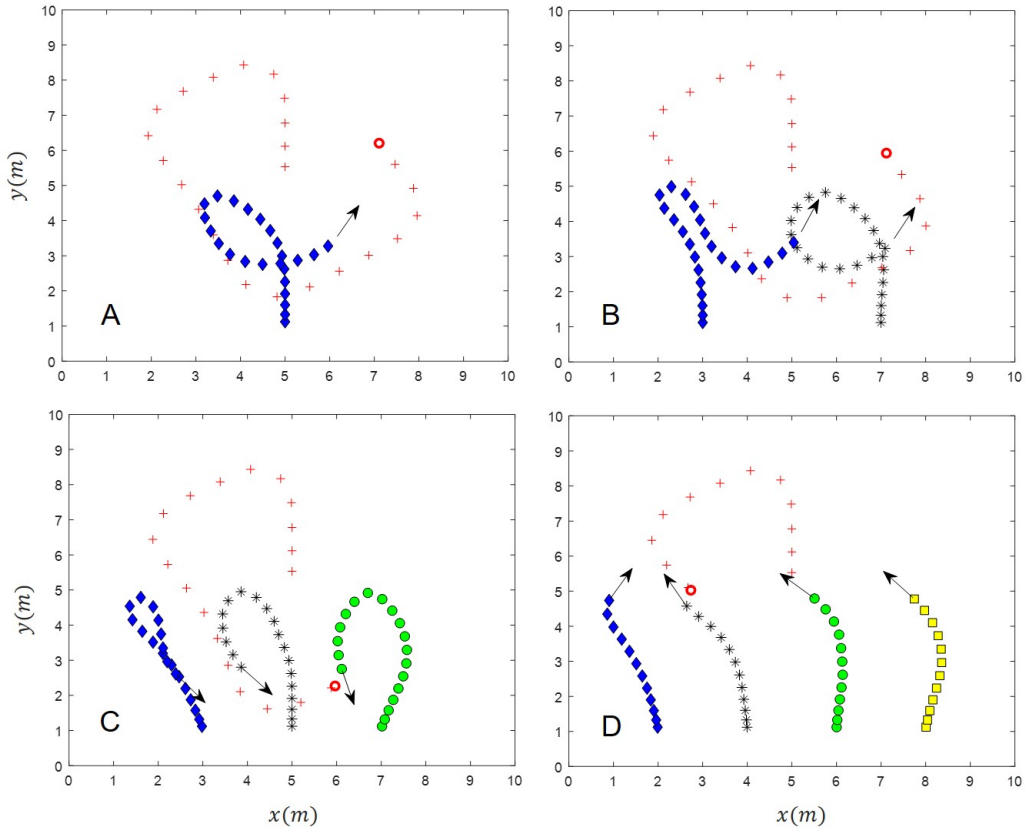


Figure 2.6 – Snapshots for pursuits with different numbers of agents (1, 2, 3 and 4). In all cases, the target does an escape trajectory-based on repulsion forces, with a velocity of 2 times bigger than the pursuers. The red circles signalize the end of the target trajectories.

In Figure 2.6, observe that in the two firsts cases (A and B), the target was not caught. Nevertheless, in the two last cases (C and D), the target was successfully intercepted before the limit of time, evidencing the importance of the increasing number of pursuers in the chase's success.

Large number of pursuers: Although not being designed for a large number of agents, the proposed algorithm can be easily scalable due to its decentralized architecture and local observation. We illustrate that in a pursuit with up to 15 pursuers, see Figure 2.7. Remark from this figure that an emergent "flocking" pattern appeared, where each agent computes independent trajectories following the target with no information of neighbors distance.

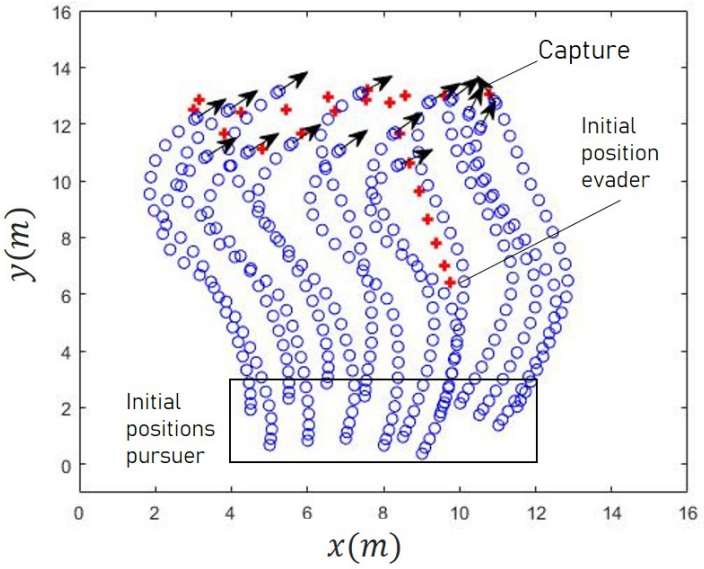


Figure 2.7 – Case of 15 pursuers and an intruder. Notice the well performance of the pursuers when tracking the target.

Effect of the offset angle: This qualitative evaluation aims to assess the effect of the offset (α_0) on the total number of catches. To this end, we subject the GDP strategy to the benchmark with seven different offset values, varying incrementally from $\pi/32$, as indicated in the legend of the Figure 2.8. More details about the benchmark setup will be given in Section 2.5

Figure 2.5-top shows the number of captures as function of the pursuers' group size. For its turn, the graph on Figure 2.5-bottom, represents the average time taken to capture.

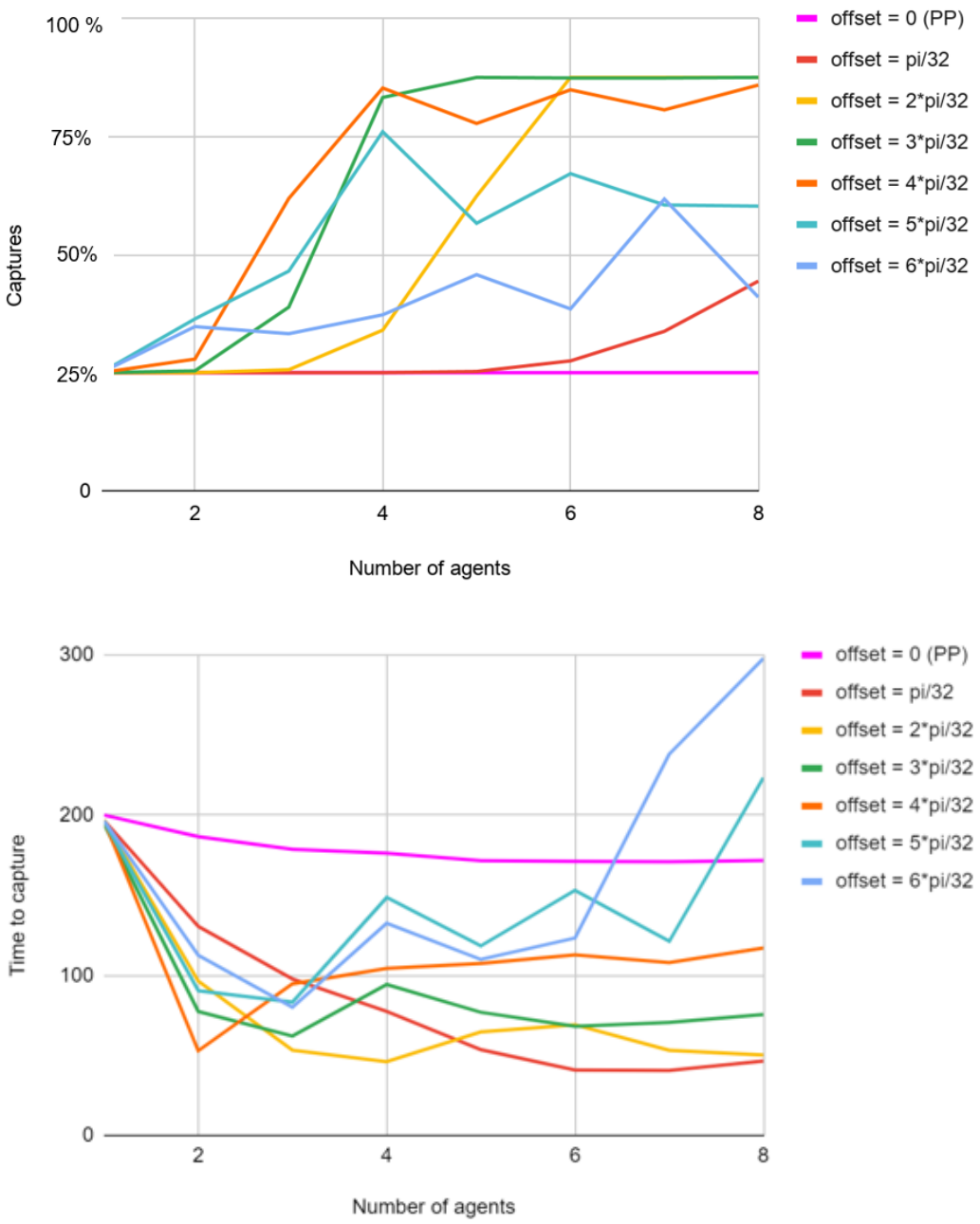


Figure 2.8 – Effects of the offset value in the pursuit.

Besides, note that the first value ($\alpha_0 = 0$) is equivalent to the classic Pure Pursuit (PP), where all agents point to the target's current position. However, applying this configuration, the cooperative behavior does not emerges, and the total catches remain practically the same during all the numbers of agents.

Therefore, with the smallest increment of offset ($\alpha_0 = \pi/32$), the size of the group starts to affect on the total amount of catches. The number of captures tends to increase, culminating in a cumulative total of 4233 catches for an offset of $\pi/4$. Please note also that for the offset value $\alpha_0 = 3\pi/32$, our strategy achieve full benchmark from 5 pursuers on, showing itself as an efficient strategy for chasing a faster evader.

For offset values bigger than $\pi/4$ the capture's performance tends to decay. This behavior was already observed in the experiments, and its justification is due to the fact that large offset values result in very open trajectories towards the target, allowing the evader to escape between two pursues.

Due to time constraints, we have not carried out additional work to find optimal parameters for this pursuit. Most likely, the use of smaller increments between the two best results ($\pi/4$ and $3 * \pi/32$) could culminate in even better results. However, we restrict ourselves to the current results, which seems to be sufficient to reaffirm the cooperative behavior emerging from the GDP strategy.

Finally, note that for the two bigger offset values ($5 * \pi/32$ and $6 * \pi/32$), although showing average yield in the total number of catches, they perform well for smaller groups of pursuers (2 and 3). This shows us that the current way of calculating the offset, which is directly proportional to the number of neighbors, tends to harm individuals at the extremities, who receive high offset values. A natural solution to this dilemma could be the use of a nonlinear function for calculating the offset, which would tend to "saturate" the offset values from a certain number of pursuers.

Effect on the collision: The graph on Figure 2.9 clearly illustrates the effect of the offset on collision avoidance. For offset = 0, collisions tend to grow exponentially, reaching values up to 52 times higher than using $\alpha_0 = 3 * \pi/32$.

The collision between pursuers is clearly undesirable; nevertheless, it is difficult to practice since all agents tend to have a common point, the evader. However, when dealing with robotic implementations, we have established a low-level safety layer that ensured collision avoidance during the experimental tests (See Experiments Section 4.2).

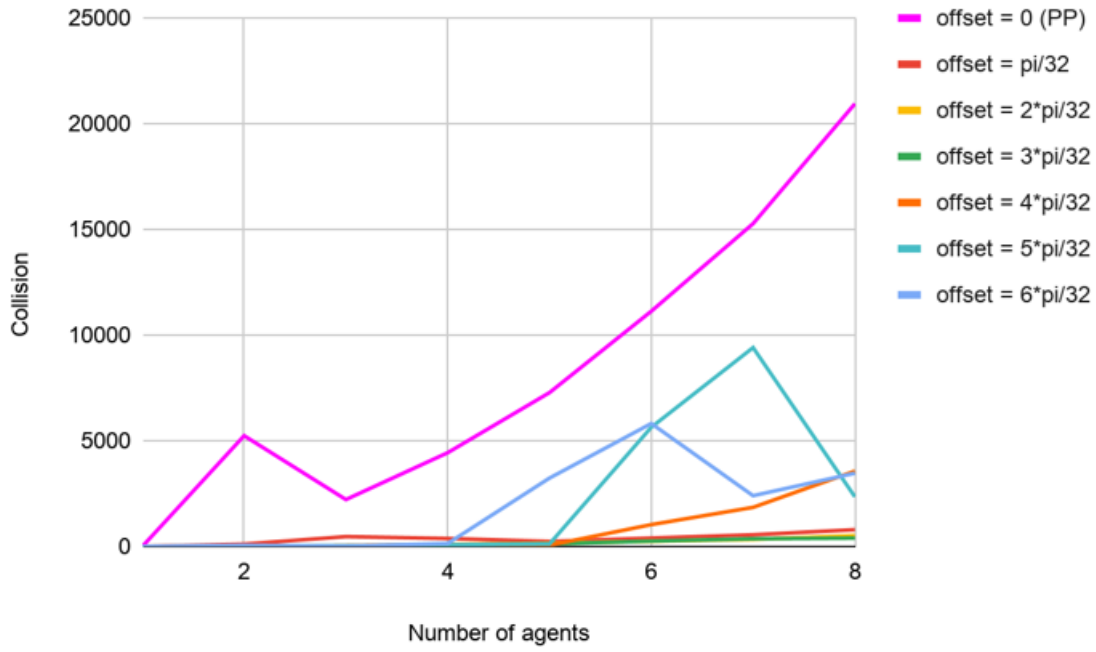


Figure 2.9 – Effect of the offset value (α_0) on the collision between pursuers.

2.3 Group Mixed Pursuit

In this section, we present our second proposal for collective hunting using guiding laws. This new algorithm seeks to go beyond ambush behavior and reproduces a more efficient pursuit, especially in the presence of faster or more agile prey. The name "mixed persecution" comes from the fact that this new GL incorporates two customary laws: the Deviated Pure Pursuit (DPP) and the Proportional Navigation Guidance (PNG).

This section's content is part of an article, still in the process of preparation, which will be entitled: **Flocking, pursuit, and avoidance: a constant bearing guidance law for multi-agents** C. de Souza, P. Castillo, B. Vidolov.

Figure 2.10 – Predators par excellence in nature use "Parallel Navigation" rule in their final approach. Left: the Falcon Peregrine, which its hunting trajectory has been described in [22]. Right: the Dragonfly, excellent insect predators that use a "camouflage strategy" to deceive their prey [164].

2.3.1 Brief preface and literature review

In the previous section, we could see the interest in using guidance laws in collective hunting algorithms. The GL was able to produce verisimilar hunting behaviors, and the use of different deviation angles was responsible for the ambushing behavior and avoiding collision at once. Although interesting behaviors were obtained, we observed that this algorithm does not perform well in the presence of a faster target.

In fact, the algorithm used is based on Pure Pursuit (PP), where the predator always addresses the prey's current position. This strategy constantly led to a "tail chase," pursuit, which ultimately gives no chance for the slowest chaser.

The disadvantages of PP towards moving targets are well known, and that is why it has been constantly replaced by Parallel Navigation (or constant bearing pursuit). This geometrical rule, apparently known since antiquity, has been extensively used by mariners to provoke an interception or even avoid a collision with other ships [15, 36]. Moreover, its principle is straightforward: if the bearing angle of two bodies is constant, and the distance is decreasing, these vehicles are on a collision course. Furthermore, this principle guarantees the optimum trajectory of the pursuer in the event of a non-maneuvering target.

This principle is so simple yet so powerful that it has been the most popular in handling missiles [36]. The family of guidance laws that implements this rule is called Proportional Navigation Guidance (PNG). The first flying bombs that implemented it were fired in the sixties, and today it still arouses interest in the scientific community in guided munitions [21, 24]. Recently, in addition to missiles, several robotic systems have used this principle for various applications, such as autonomous landing [41], navigation in a clustered environment [1], and intruders' interception [28].

Furthermore, with the recent image acquisition technologies, scientists have identified this principle as the attack behavior of several predators, such as the peregrine falcon [22] and some insects [19]. In this context of natural predation, the proportional navigation law (PNG) is confused with the concept of camouflage laws, where the predator moves towards the prey so that it sees it "static" to the background [124].

Its importance is evidenced by the applications and observations above; however, Parallel Navigation presents certain disadvantages mainly in the final stage and through maneuvering targets (changing directions abruptly). Besides, it requires high values of lead angle when the speed ratio is low (which can be quite common in the presence of a more agile prey) and can provide loss of sight of the target, in the case of a field of view sensor limited [36].

To minimize these disadvantages, Shneydor, in [36], devised a "mixed" guidance law of between "Pure Pursuit" and "Parallel Navigation", since this first one, although less efficient, does not suffer from the low-speed ratio.

With similar motivation, we propose in this work a mixed guidance law, composed of two terms, one from PN and the other from DPP. The first one is desired to be predominant in the pursuit, since it has a more efficient trajectory. In turn, the second term has dual functionality: first, ambushing (as explained in the previous work), and second limiting the lead angle of the pursuer to target, preventing the loss of sight of the evader.

2.3.2 Group Mixed Pursuit (GMP) algorithm

Considering the evolution model exposed in (3.16), let us consider the following guidance law, which composed of two terms, one for the Proportional Navigation (f_{png}) and the other for the Deviated Pursuit (f_{dpp}). Thus the mixed GL will have the following format:

$$f_{\psi_i} = \sigma_a(f_{png}) + \sigma_b(f_{dpp}) \tag{2.6}$$

where $\sigma_x(.)$ is a saturation function that can be defined by

$$\sigma_a(x) = \begin{cases} a, & \text{if } x > a \\ x, & \text{if } a \geq x \geq -a \\ -a, & \text{if } x < -a \end{cases} \tag{2.7}$$

Therefore, the threshold of σ_b must be bigger than σ_a, to allowed the second term to overcome the first. This is necessary since f_{dpp} is responsible for shot-range collision avoidance between two pursuer. In the next, we present the composition for both terms.

Proportional Navigation term

A simplistic GL, which implements the Parallel Navigation geometrical, can be defined as follow:

$$f_{png} = K_{pn} * \dot{\lambda}_{iT} \tag{2.8}$$

where K_{pn} is a positive gain. This GL is also called Proportional Navigation Guidance (PNG), and it is described in Chapter 1.

As stated in the introduction, the PNG carries the drawback of provoking a large leading angle $(\psi_i - \lambda_{iT})$ from the pursuer to target when the ration (v_i/v_T) is low, which can cause loss of sight of the target, considering limited field-of-view of the pursuer.

Deviated Pursuit term

In order to avoid the target's loss of sight, which can be a non-desired effect of the PNG, we proposing the following modified DPP law:

$$f_{dpp} = K_{dpp}(d_{min}) * \tan\left(\frac{\psi_i - \lambda_{iT} - \alpha_i}{\beta * \pi^{-1}}\right) \tag{2.9}$$

where β is the opening angle of the pursuer (or perception sensor). The $tan(.)$ function was chosen for being odd-symmetric and not defined in $\psi_i - \lambda_{iT} = \beta$. Thus this term provides asymptotic values when the heading error is close to the limits of the field of vision β.

Furthermore, we introduce the gain $K_{dpp}(.)$ that is inversely proportional to the smallest distance between pursuer d_{min} and is null when the distance is bigger than a threshold R_0. It can be computed as follow:

$$K_{dpp}(d_{min}) = \begin{cases} K_0 * \left(\frac{d_{min} - R_0}{R_0}\right)^2, & \text{if } d_{min} < R_0 \\ 0, & \text{else} \end{cases} \tag{2.10}$$

As a result, the DPP term has a short-range performance, being canceled when two pursuers are not on collision imminence; and evidenced by the proximity of two pursuers.

The mixed GL

The first term of (2.6), corresponding to PNG law, is desired to be the predominant during most of the pursuit since it provides a more efficient trajectory. Nevertheless, there are two situations where the DPP term should overcome the PNG: first in the eminence of collision with the closest neighbor ($d_{min} < r_0$), and second, when the target is getting close to the limit of the field of vision, i. e., $|\psi_i - \lambda_{iT}| \to \beta$.

2.3.3 Numerical validation

PNG vs PP

In the scenario illustrated in Figure 2.11, we intended to give the reader unfamiliar with guidance laws an expository comparison between the two methodologies covered in this section. Therefore, to illustrate the advantage of the Proportional Navigation Guidance (PNG) over the Pure Pursuit (PP), the following scenario is proposed: two identical pursuers are located initially in the bottom coordinates $(x, y) = (2, 0)$m; while a target (red) starts from the position $(x, y) = (-8, 10)$m.

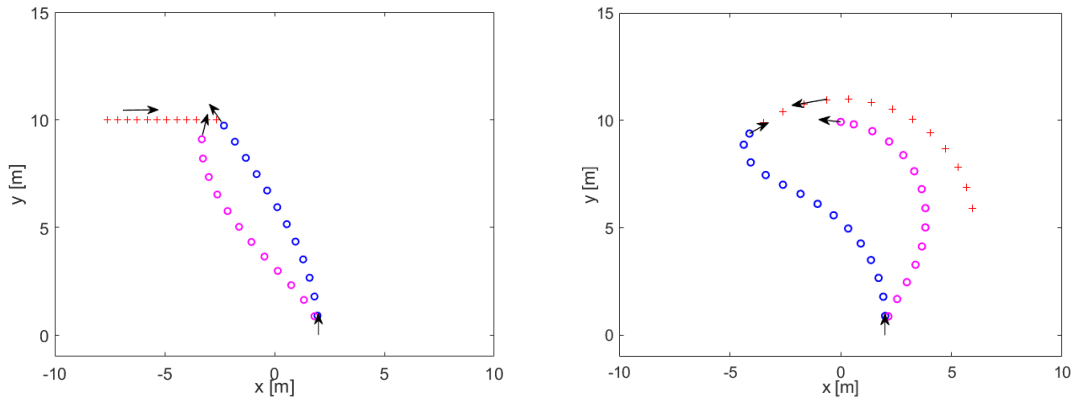


Figure 2.11 – Comparison between Pure Pursuit (PP) and Proportional Navigation (PNG). The PNG pursuer (blue circles) and the PP pursuer (pink circles) are not aware of each other, i.e, there is no cooperation. In the left picture, the target performs a linear trajectory, while in the right one, the target executes a maneuvering trajectory.

The first pursuer (pink circles) implements a PP algorithm while the second (blue circles), the PNG algorithm. In Figure 2.11-left, observe the pursuer's performances when tracking a target with $v_i = 2 * v_T$; note that PNG algorithm provides a shorter path and prevails at the end of each course.

In Figure 2.11-right, a similar scenario is illustrated; nevertheless, the target does not have linear movement. Notice that even if the algorithm is not designed for non-maneuvering targets (red crosses), the PNG (blue circles) succeeds and provides a shorter path than the pure pursuit (pink circles).

Mixed Pursuit vs Deviated Group Pursuit

In this second example, we will make a qualitative comparison between the mixed pursuit (2.9) and the methodology presented in the section 2.2.2, the Deviated Group Pursuit (DPG) (2.5).

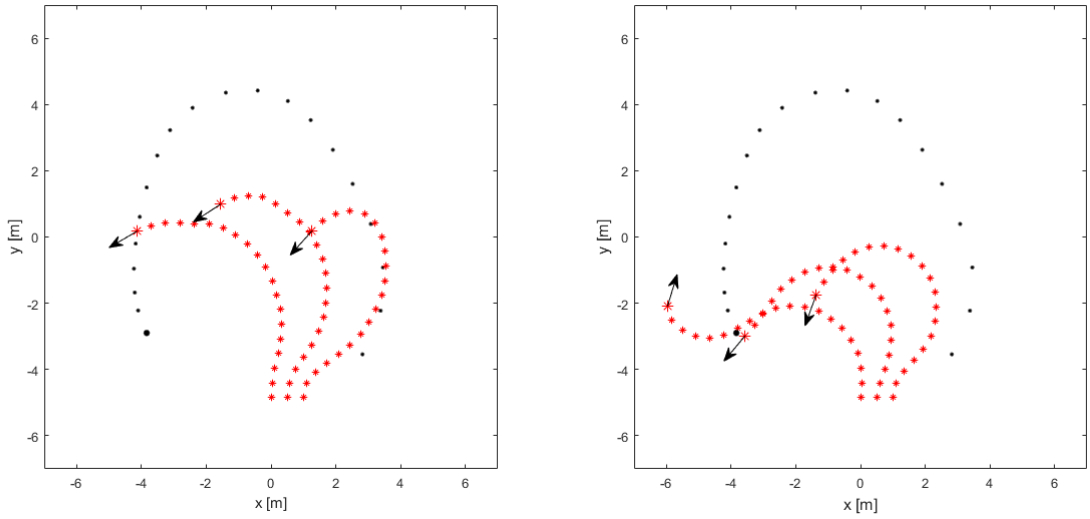


Figure 2.12 – Comparison between Deviated Group Pursuit (DGP, left picture) and Mixed Group Pursuit (MGP, right picture). In both cases the target is navigating in the same trajectory.

For this, consider the scenarios shown in Figure 2.12: Three pursuers (red stars) are initially located at the bottom of the arena, positions (0, -5) (0.5, -5) (1, -5)m, moving with fixed speed of 0.45 meters/frame. The target performs a maneuvering trajectory with variable speed, but that is up to 3 times higher than the pursuer speed. The target behavior was obtained from recording experimental trajectories with a quadcopter.

Still, in Figure 2.12, the left image represents a group chase where the agents implemented the DPG. Note that even though the ambush formation is created, the trajectory is not efficient, and the target is not captured in the course of this screenshot. As previously discussed, this is since the pursuit is directed towards the target's instantaneous position, and no consideration is given of its dynamics.

In Figure 2.12-right, we can see that the target is captured, and the agents still kept an ambush formation. Note that the "straight line condition course," the geometric rule behind PNG, promotes an efficient path even when it comes to maneuvering trajectories.

Effect of the offset: Similar to the previous section, we submitted the GMP to the pursuit benchmark, considering the same offset intervals. The results regarding the number of captures and average time can be seen in the respective Figure 2.13.

Note that the same performance patterns presented in GDP were repeated for this strategy. Firstly, the best results in general terms are for the offset values between $3 * \pi/32$ and $4 * \pi/32$. Secondly, the highest offset values, $5 * \pi/32$ and $6 * \pi/32$, have relatively high capture values for 2 and 3 pursuers, but their yield drops as the number of pursuers increases.

These results reinforce the conclusions of the previous section. Besides, note that, unlike the last section, the results for offset = 0 do not correspond to the classic PNG navigation law. This is due to the tangent term (Eq. 2.9), which restricts the pursuer's heading movement, differentiating it slightly from the classic implementation of PNG.

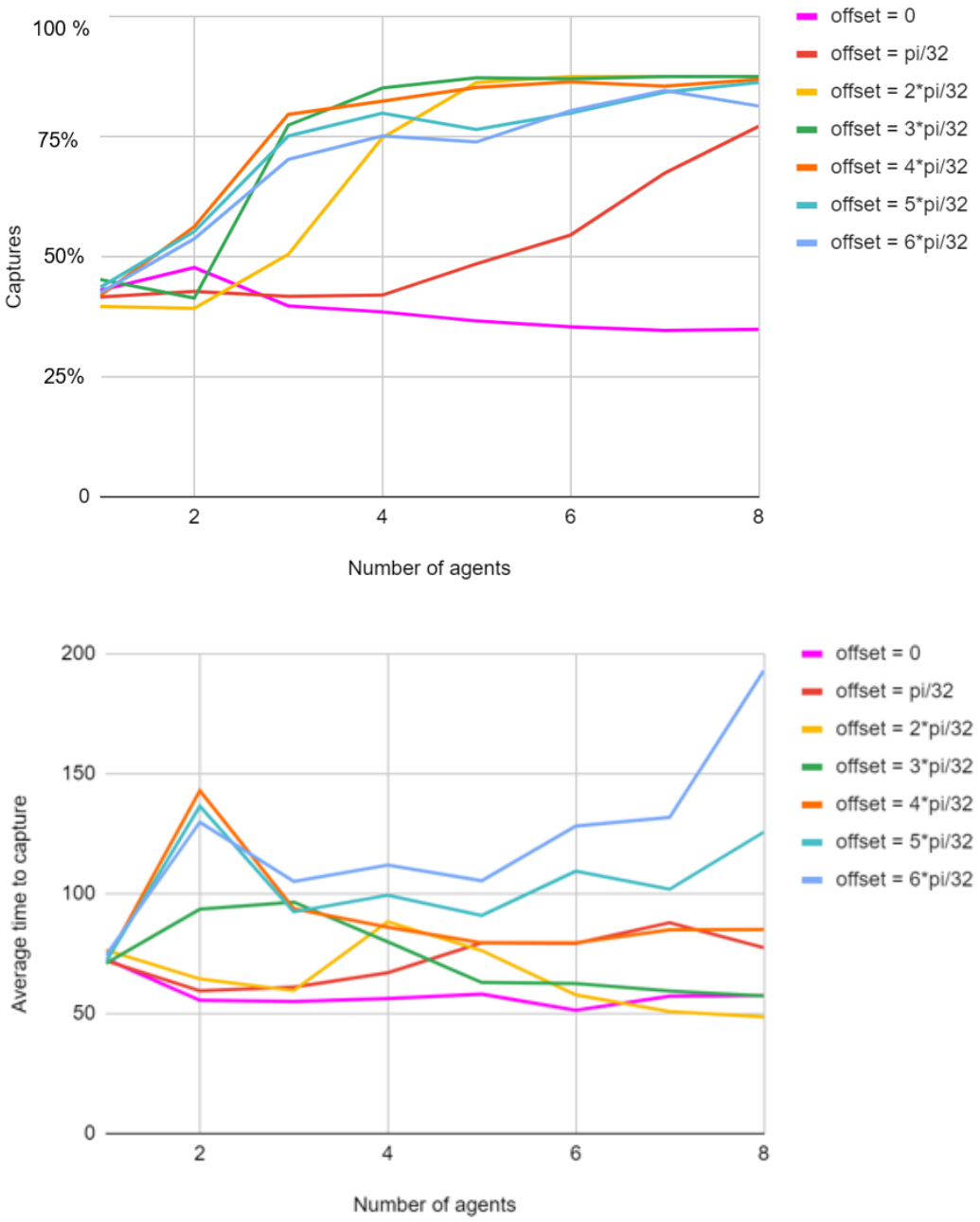


Figure 2.13 – Effects of the offset value in the pursuit.

2.4 Reinforcement Learning Pursuit

In this section, we propose an approach for multi-agent pursuit using Deep Reinforcement Learning (Deep RL). Henceforth, instead of using guidance law established to model the pursuit, the pursuer behavior is controlled by a neuronal network policy. To determine the policy, we considered a multi-agent environment, bounded and using the kinematics equations (2.1). In this, the purser will "learn" its behavior by trial-and-error interaction in a simulated environment.

This work was done in collaboration with the Robotic Centre of Monash University, Melbourne, Australia; during an institutional exchange, funded by the UTC doctoral school and the Heudiasyc laboratory. This section's content has been recently organized in a paper entitled: **Distributed Multi-Agent Pursuit using Deep Reinforcement Learning**. C. de Souza Jr, R. Newburry, A. Cosgun, P. Castillo, B. Vidolov, D. Kulić. Accepted to ICRA/RA-L 2021.

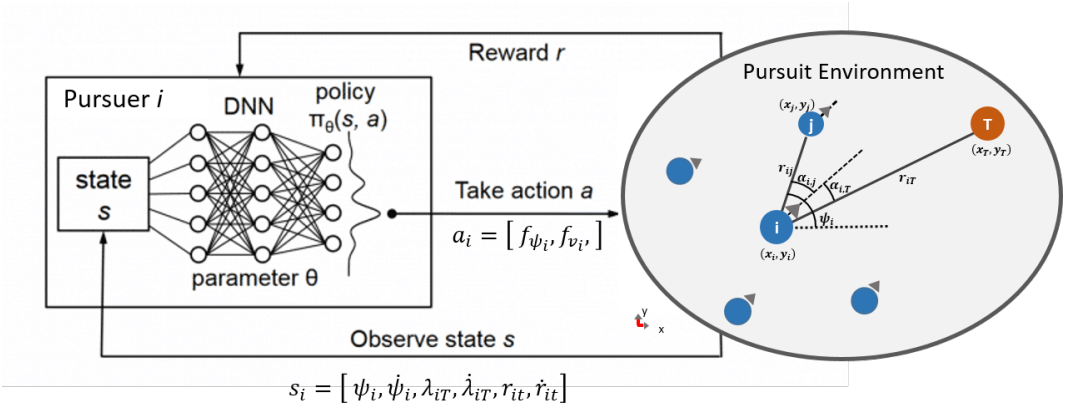


Figure 2.14 – Scheme for the Deep Reinforcement Learning.

2.4.1 Brief preface and literature review

Until now, all pursuit strategies have been designed in a bottom-up manner; the microscopic relationships have been previously established, and then the emergent behavior was analyzed. Nevertheless, in this section, we will use an automatic behavior design technique by applying Deep Reinforcement Learning.

In this, the purser will "learn" its behavior by trial-and-error interaction in a simulated environment.

In the last decade, with the advances and popularization of Deep Learning, another recent technology has stolen the spotlight from the robotic community, Deep Reinforcement Learning. These advances allowed learning of complex tasks in continuous environments, which has increasingly enabled Reinforcement Learning in robotic applications, such as driving [155], grasping [156] and manipulation [159].

Deep RL has also proved to be a powerful tool for the complex task of multi-agents pursuits, such as in [92, 93, 94, 105]. However, most of these works consider models and hypotheses very distant from real-world applications, such as local measurement and locomotion restrictions.

In this work, we propose a multi-agent pursuit approach against a faster evader. We consider it a decentralized system, where the pursuers decide their own actions based on local observations of the environment. The perception of the target and the neighbors are given in local and polar coordinates, which are consistent with commons embedded sensors for autonomous navigation, such as camera and lidars.

We use Twin Delayed Deep Deterministic Policy Gradient (TD3) [108], a state-of-the-art deep reinforcement learning algorithm that was successfully applied to other domains[160, 161], with a state representation that encapsulates relative positional information of neighboring agents as well as the target, and use a group reward structure that encourages good formations. During training, curriculum learning is applied to start with an easier version of the problem, and gradually learn the task with increasing difficulty.

Related work

The pursuit-evasion game is a highly studied task in multi-agent RL [95, 96, 97]. However, most approaches apply only to omni-directional agents, which cannot be easily transferred to real robotic applications without a loss in performance.

Lowe [94] presented an approach for multi-agent reinforcement learning using an adapted version of an actor-critic algorithm which they extended to multi-agents. They use their approach on the pursuit-evasion game with omni-directional agents and showed that it outperformed Deep Deterministic Policy Gradient (DDPG) [86] on the same scenario. Xu [93] considered pursuit-evader games with non-holonomic agents, where new agents can join the game. They adapted Bi-directional Recurrent Neural Networks [163] and DDPG, however, they only consider a situation with 3 and 5 agents. Furthermore, the observation also assumes global information about other agents which limits the applicability to real world situations. Hüttenrauch [92] studied multi-agent pursuit evasion systems by considering the agents as a swarm. They consider agents in the swarm as interchangeable and the exact number irrelevant. They create a new state representation based on mean embedding of distributions. Their work focuses on scalability and shows that their system can operate with up to fifty agents.

Curriculum Learning Curriculum learning [102] is a learning paradigm to improve the speed of convergence and reduce local minima by gradually increasing the complexity of training data. This learning paradigm has been been widely used for RL[109, 162] and deep learning [107]. Gupta [105] used curriculum learning in multi-agent RL to solve problems which were previously considered intractable. They show that Trust Region Policy Optimization (TRPO) [158] has a better performance compared to Deep Q Networks (DQN) [87] in the discrete domain and DDPG in the continuous domain.

Our work, while borrowing ideas from both classical methods and learning-based methods, focuses on using DRL to improve the pursuit performance and consider operational metrics such as capture success rate and the average time to capture. Furthermore, we propose a method that is suitable for sim-to-real policy transfer with realistic observation models and non-holonomic constraints. To our knowledge, we are the first to demonstrate a real-world multi-agent pursuit implementation using a DRL policy.

2.4.2 Methodology

Multi-Agent Deep RL: We consider our task to be a Markov Decision Process (MDP) defined by tuple $\{\mathcal{S}, \mathcal{A}, R, \mathcal{P}, \gamma\}$ where $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$, $r_t \in R$ are state, action and reward observed at time t, $\mathcal{P}$ represents an unknown transition probability from s_t to s_{t+1} taking a_t, and γ stands for the discount factor. The reinforcement learning goal is to maximize the sum of future rewards $R = \sum_{t=0}^{T} \gamma^t r_t$, where r_t is provided by the environment at time t. For continuous control, actions are sampled from a deep neural network policy $a_t \sim \pi_\theta(s_t)$, where a_t is the angular velocity of an individual pursuer.

Our approach is based on distributed, in-series learning with parameter sharing. For learning, we use Twin Delayed Deep Deterministic policy gradient approach (TD3) [108] using an experience replay buffer. The agents are considered homogeneous, allowing sharing the same policy and parameters, although the execution remains decentralized.

State-representation: The observed states of a pursuer i for a total of n pursuers $s_i = [\psi_i, \dot{\psi}_i, s_{iT}, s_{i1}, s_{i2}, .., s_{in-1}]$ where ψ represents the heading with respect to a fixed world frame, s_{iT} denotes the state of the target relative to pursuer i and s_{ij} defines the state of pursuer j relative to pursuer i. The relative state of the target with respect to pursuer i is $s_{iT} = [r_{iT}, \dot{r_{iT}}, \alpha_{iT}, \dot{\alpha_{iT}}]$ and the relative state of the pursuer j with respect to pursuer i ($i \neq j$) denotes $s_{ij} = [r_{ij}, \alpha_{ij}]$, where r_{ij} is the Euclidean distance between pursuers i and j and α_{ij} represents the heading error defined in local an polar coordinate frame attached to pursuer i.

In order to generalize the learning, the neighbors states s_{ij} are sorted with respect to their proximity to the center of the heading of the agent i, i.e., the first agent $j = 1$ is the one with smaller heading error $\alpha_{ij} = \psi_i - \lambda_{ij}$. Another generalization is limiting the number of observed neighbors (N_O) and filling up this sub-group with the N_O closest pursuers. This strategy allows the scalability of the system.

Reward structure: At each time step, each agent individually receives a reward designed to incentivize the evader's capture and encourage an adequate formation of pursuers. The reward function is:

$$Re_i = \begin{cases} Re_{captor}, & \text{if } r_{iT} \leq R_{cap} \\ Re_{helper}, & \text{if } r_{iT} \leq d_{cap}, \exists j \neq i \\ -w_q\, q - w_d\, r_{iT}, & \text{otherwise} \end{cases}$$

At each step that the target is not captured, every agent receives a negative reward that is a linear combination of an individual reward (its distance to the target r_{iT}) and a group reward (**q score** proposed by [106] which encourages the agents to make a good formation). The **q score** is a scalar number in the range $[0, 2]$, which provides a metric for evaluating the fitness of the formation of the pursuers (lower is better). Also, we penalize r_{iT}, which helps encourage the agents to get close to the evader while being in a good formation. The weights w_q and w_d can be chosen such that when the agents are close to the target, the reward is dominated by the **q score**, encouraging good formation. Therefore, the formation score (q) is defined as:

$$q = \frac{1}{n} \sum_{i=1}^{n} (r_{\mathbf{0T}} \cdot r_{\mathbf{iT}} + 1) \tag{2.11}$$

where $\mathbf{r_{iT}}$ represents the normalised vector between agent i and the target, and n is the number of agents. In this equation, the closest agent to the target is defined as an agent 0. The formation score encourages agents to spread around the target (i.e., approach the target from different directions). Early experiments with the formation score as part of the reward function showed that when the formation is the only component of this reward, pursuers would only form a good formation but would not attempt to capture the evader. To avoid this situation, we penalize the distance to the target, which helps encourage the agents to get close to the evader while being in a good formation. The weights w_q and w_d were chosen such that when the agents are close to the target, the reward is dominated by the formation score, encouraging good formation.

However, when the agents are far from the target, the reward will be dominated by the target's distance, encouraging the agents to move closer to the evader. We analyze the effect of the formation score on pursuit performance in Sec. 2.4.3.

If the target is captured at a time step, then the pursuer who captures the evader receives the reward r_{captor}, while the rest of the agents receive r_{helper}, such that $r_{captor} > r_{helper}$. This encourages the pursuers to act selfishly and go for the final capture while encouraging collaboration.

Curriculum Learning We apply a curriculum for learning by starting from a more accessible version of the task and gradually increasing the difficulty until the actual difficulty is achieved. There are two main factors in determining the difficulty of a pursuit-evasion game: the target's relative speed with respect to the pursuers and the capture radius d_{cap}. We vary the capture radius by starting from a large radius (so it is easier to capture the target), then gradually making it smaller. This encourages agents to not adopt a straightforward chasing tactic at the beginning of learning but to form more sophisticated behaviors, which could be transferred to smaller capture radii. We also experimented with reducing the pursuer speed to reduce the task's difficulty; however, we found that pursuers mostly learned to follow the evader directly, and it was harder to explore more sophisticated behaviors afterward.

Curriculum learning helps exploration, especially during the early stages of learning because early on, it helps the pursuers to capture the target, which would take a longer time in the actual and more difficult scenario. This helps alleviate the sparse reward problem, which is a well-known challenge in reinforcement learning [103]. We analyze the effect of curriculum learning on the pursuit performance in the subsection 2.4.3.

2.4.3 Numerical results

We use the following simulation parameters: Arena radius $R_{arena} = 430$ pixels, $T_{timeout} = 500$ iterations, rewards r_{captor}, r_{helper} and the weights w_q and w_d were set to 10, 100, 0.1, 0.002 respectively. The capture radius R_{cap} was set to 30 pixels for testing, although this value was varied as part of curriculum learning. For fixed linear velocity cases, the pursuers' speed v_p was 10 pixels per timestep, while the target's speed v_T varied from 0 to 20 pixels per timestep. The maximum angular rate ω_{max} was fixed at $\pi/10$ per timestep. The numbers of pursuers n varied between 1 and 8, and they were initialized at random positions within a circular area with radius of 100 pixels around the center, while the evader was initialized at a random position outside the circular area with a radius of 300 pixels. The rest of this section covers the evader behaviors and baseline methods.

Effect of Curriculum Learning

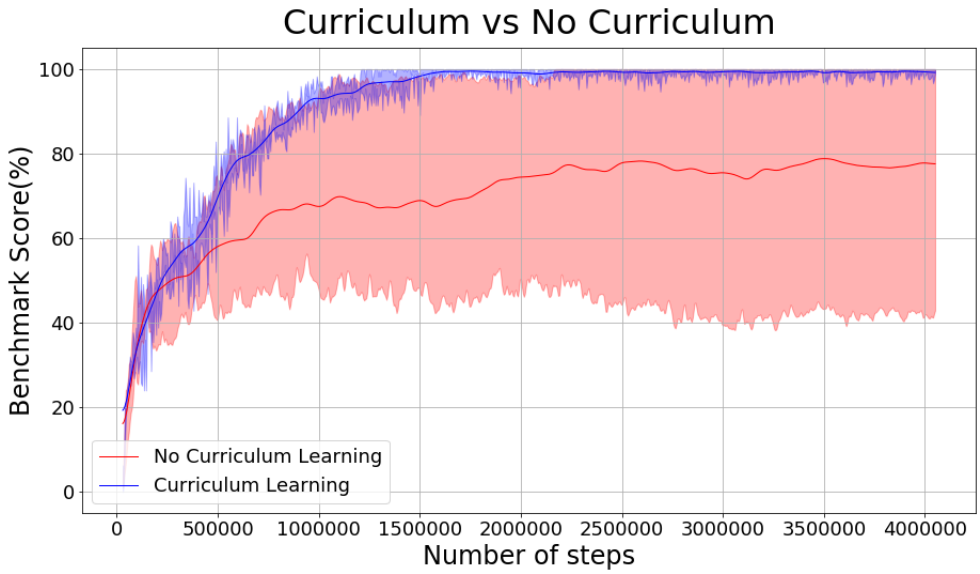


Figure 2.15 – Comparison of capture success rate with and without curriculum learning, with respect to the number of training steps. With curriculum learning, the benchmark scores are much higher and more consistent.

Figure2.15 compares the effect of using our curriculum learning strategy described in subsection 2.4.2, for $n = 3$ agents. The network was trained three times, and during training, we analyze the evolution success rate trained policy. At regular intervals, we stop training and evaluate the policy on the repulsive evader benchmark. The results show that curriculum learning is very helpful for capture performance: it converges to about 100% success rate after 1.5 million training steps, whereas without curriculum learning, the average success rate was below 80% even at 4 million training steps. Furthermore, curriculum learning performance was much more consistent, as evidenced by the low variance among the three runs.

Effect of formation score in reward function

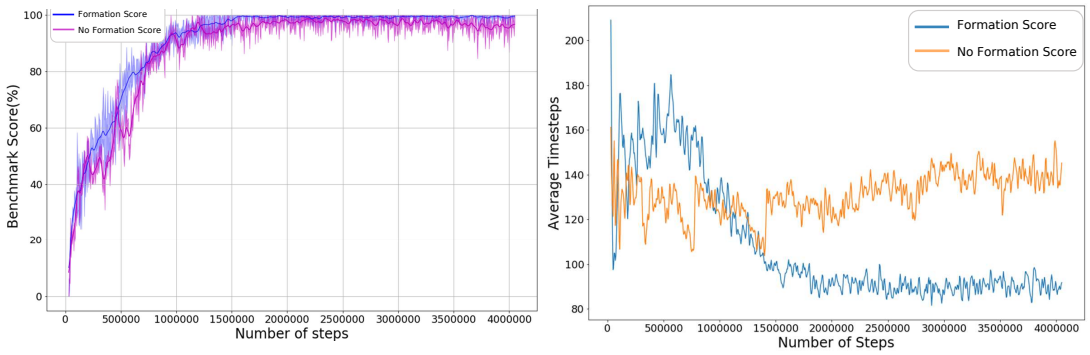


Figure 2.16 – Using a formation score as a dense reward results in more captures, in less number of timesteps on average.

We provide a partial reward at every timestep in order to encourage good formations. We analyze the effect of supplying this dense reward component to each agent. Fig. 2.16 compares the evolution of the capture performance with and without the formation score, with respect to the number of training steps. These experiments were conducted with $n = 3$ agents. As can be seen in the figure, benchmark scores were slightly higher when the formation score is used as part of the reward. Furthermore, when the formation score is used, the average capture time for successful episodes is decreased.

Qualitative analysis of emergent behavior

We observe two interesting learned emergent behaviors that often leads to successful captures: ambushing and splitting up.

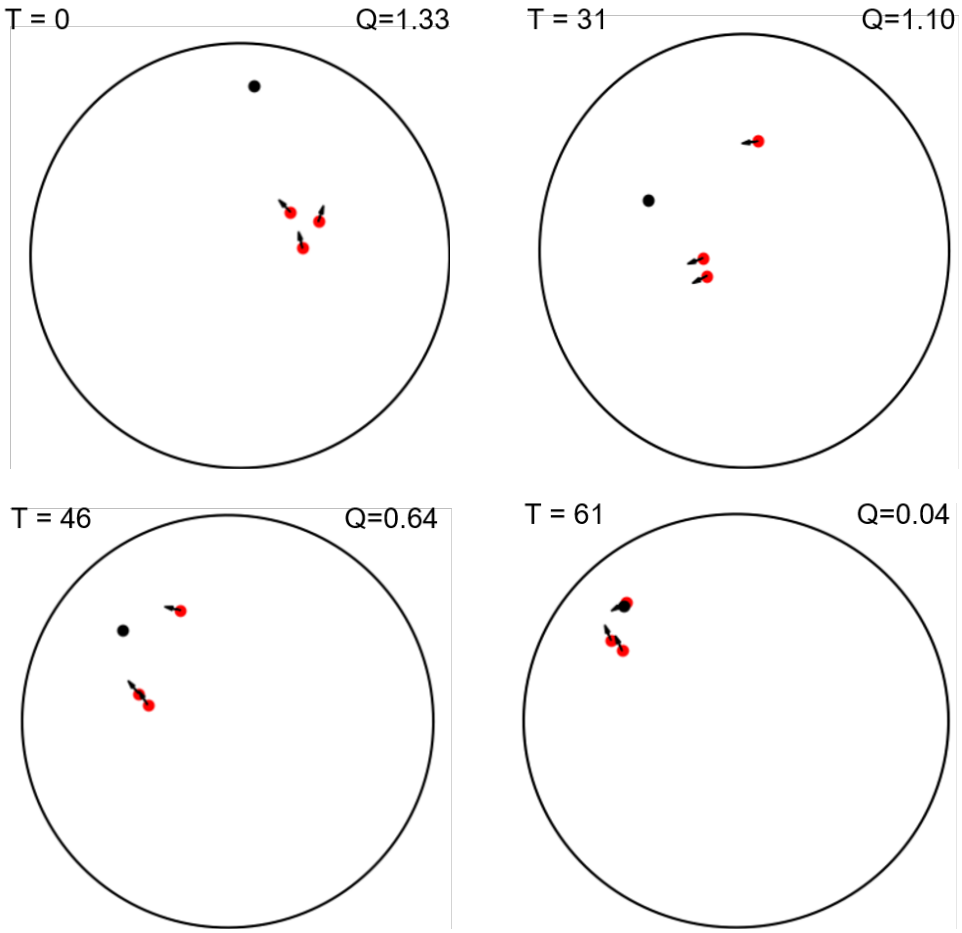


Figure 2.17 – "Split Up" strategy learned by 3 pursuers. Timestep (T) and formation scores (Q) are shown at four snapshots. The target is shown as the black circle. The agents start in a random direction (Top Left), push the agent towards the wall (Top Right), split into two groups (Bottom Left) before going for the capture (Bottom Right).

Fig.2.17 shows the splitting up behavior with $n = 3$ agents. This behavior was more common with a smaller number of agents. The agents tend to split up in to two groups, trying to push the evader in to a wall before attempting to block the two opposite directions.

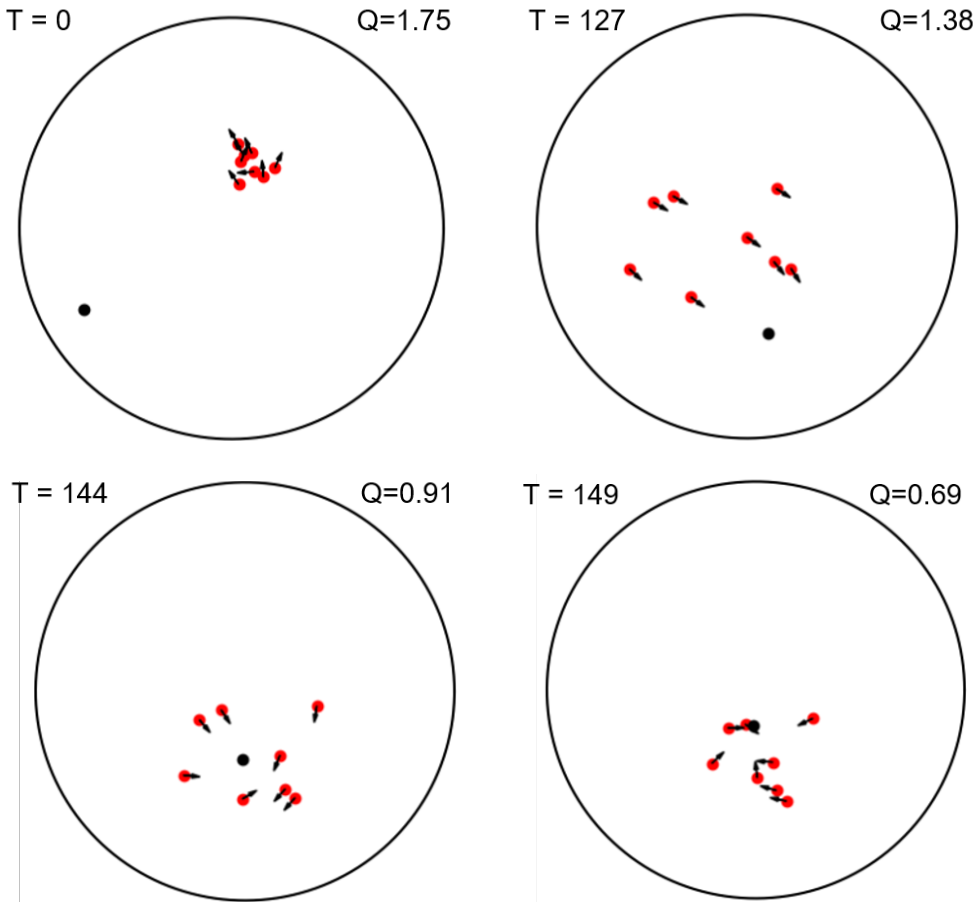


Figure 2.18 – "Ambush" strategy learned by 8 pursuers. Timestep (T) and formation scores (Q) are shown at four snapshots. The target is shown as the black circle. The agents start in random directions (Top Left), move as a circle (Top Right), ambush the target (Bottom Left), and capture it (Bottom Right).

Fig.2.18 shows the ambushing behavior with $n = 8$ agents. This behavior was more common with a larger number of agents. The agents tend to form a circle, attempting to move such that they can completely surround the evader and then approach from all directions. This is similar to pack behaviors observed in Muro's work [33]. When the agents execute this strategy to trap the evader, it is often very difficult for the evader to escape.

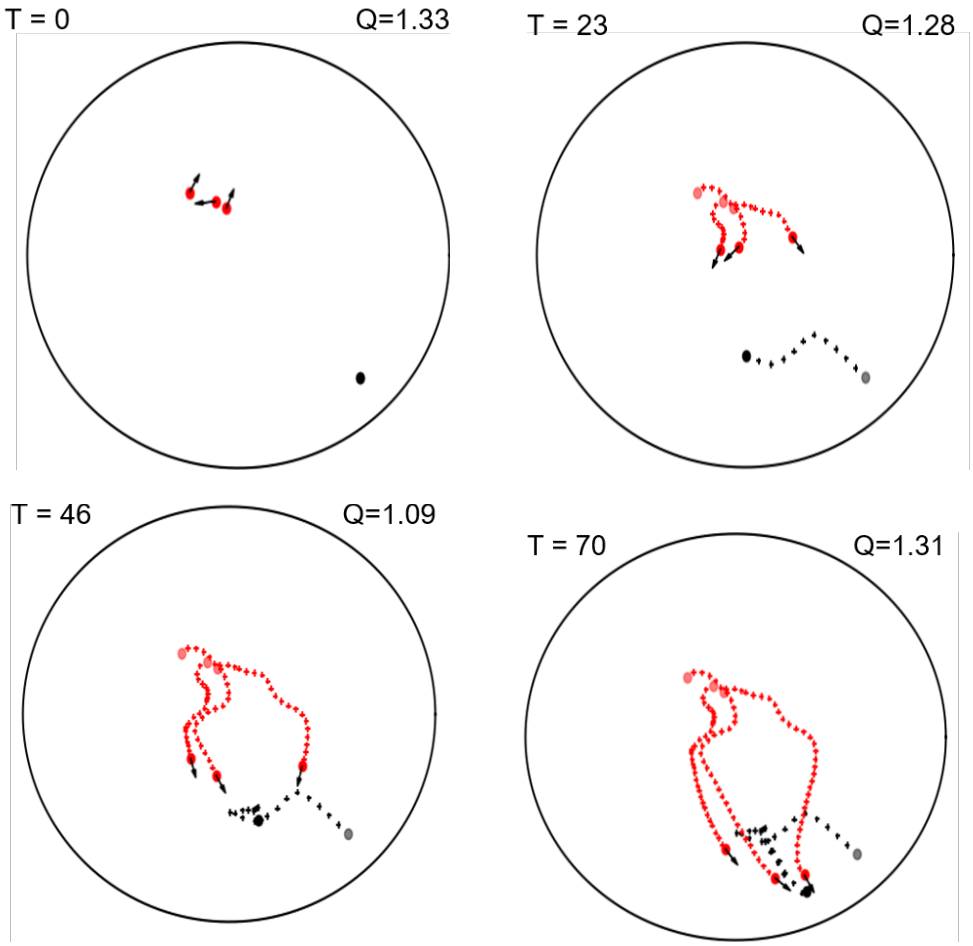


Figure 2.19 – "Stalking" strategy learned by 3 pursuers with velocity. Timestep (T) and formation scores (Q) are shown at four snapshots. The target is shown as the black circle. The agents start in random directions (Top Left), move towards the target (Top Right), slow down and angle themselves such that they can surround the target (Bottom Left), and capture it (Bottom Right).

Variable linear velocity

Previous sections considered constant linear speed and variable angular speed for pursuers, primarily because this is an assumption for classical algorithms. We now consider the more general case, where agents can also vary their linear velocity between 0 and v_p. Therefore, we train the network with two outputs: linear and angular velocity. We consider the 3 agent scenario, in which the agents achieve 100% capture rate with and without velocity control, in both the fixed,

and reactive benchmarks.

The agents often displayed a “Stalking" strategy as illustrated in Figure 2.19. With this strategy, the agents move towards the target, before slowing down and waiting, until the opportunity presents itself to attempt to capture the evader. This behavior may have analogues in nature, where pursuers will stalk their prey and position themselves such to maximise the likelihood of attack[45].

2.5 Qualitative comparison

In this section, we propose to evaluate the performance of our propositions quantitatively, not only among themselves but also with three baseline methods of the state-of-the-art. Two of them are based on classic multi-agent strategies, Angelani [34] and Janosov [52], where traditional flocking self-propelled agents are adapted to carry out a pursuit. The third one, Hüttenrauch [92], is part of an interesting strategy for applying deep reinforcement learning in swarming problems. Therefore, we conduct the following performance analysis: effect of the number of pursuers, arena size, relative evader speed.

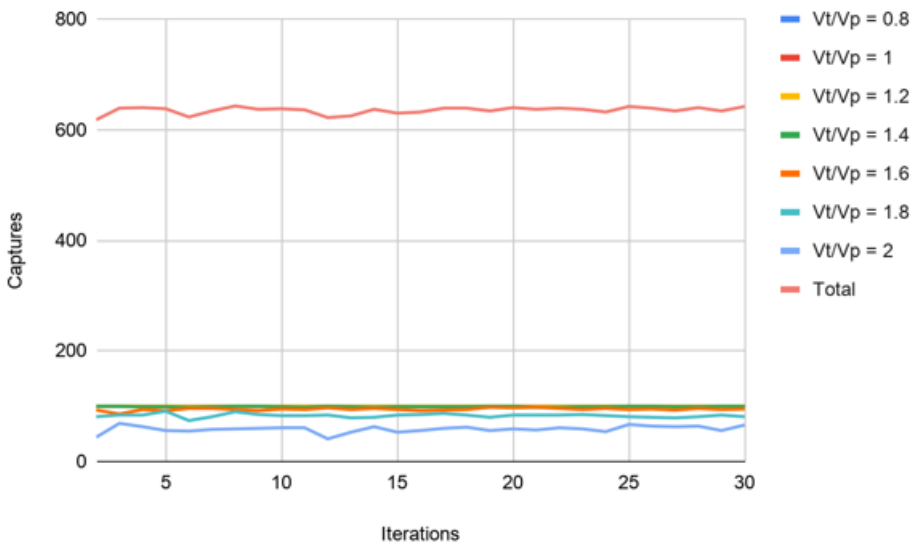


Figure 2.20 – Detailed benchmark results for 3 pursuer over 100 repetitions.

2.5.1 Benchmark

To assess our propositions' performance qualitatively, we have established a benchmark for pursuit, which will be reported below. A group pursuit environment was implemented in python, where pursuers and evaders evolve in the 2D plane. The navigable area is limited by a circular border with a variable radius. However, the standard size of the arena is $R_{arena} = 430 pixels$.

The initial positions of the pursuers and the target are initiated randomly. However, to avoid initial conditions with the target already at a disadvantage, two different initialization areas are determined. The pursuers are started within a circle of radius equal to 100 pixels, concentric with the arena. The evader is initialized outside of another concentric circle, with a radius of 300 pixels.

The standard pursuit evaluation consists of 700 episodes, where the relative speed of the target to the pursuers varies incrementally every 100 episodes, i. e., $v_T/v_p = 0.8, 1.0, \ldots, 2$. An episode lasts until captured by one of the pursers or until the limit of interactions 500 is reached.

For most scenarios, pursuers and target move at constant speeds. However, while the target moves in an omnidirectional manner (and with constant absolute speed), the pursuer moves according to the Equation (2.1), in its discrete form.

Through this benchmark, the following metrics are being calculated:

- The number of captures: which is equivalent to the number of successful episodes (Max 700). In this work, the capture radius was set to $R_{cap} = 30$ pixels.
- The average time of capture: which corresponds to the average of interactions needed to capture the target.
- Collisions: the number of times that two pursuers had their relative distance below a collision threshold, $R_{col} = 30$ pixels.

Besides, to test the benchmark repeatability, we propose the following experiment: the same pursuit scenario, with 3 pursuers implementing the GDP, were repeated 30 times. The total value of the catches is shown in Figure 2.20. As expected, the variability occurs more at high relative speeds, reaching its apex at Vt/Vp = 2. However, the total catch's standard deviation is 6.36, which corresponds to a variation of 0.90% in the captures.

2.5.2 Effect of the number of pursuer

The number of captures and the average time of six pursuit strategies are shown in Figure 2.21. Our RL pursuit strategy surpasses all others presented, reaching a complete benchmark for all agent numbers from 3 onwards. The other best results are respectively GMP, Janosov, Hüttenrauch, GDP and Angelani.

Although the GDP has a slightly lower result than Huttenrauch and Janosov, it is still an interesting option, especially when considering its minimalistic implementation. Besides the GDP computational simplicity (Equation 2.5), it is the single one possible to implemented by using only bearing information for the whole pursuit. Which gives you a good advantage in practical terms since it can be implemented using only an RGB camera.

GMP fulfills its role of improving GDP, and its results are clearly more performant. Although more information is required for carrying the pursuit in GMP, it is still compatible with the implementation in real systems. As shown in equation (2.9), this strategy requires, besides the bearing, the relative distances, and the bearing angle rate towards the target. Although more hardware is needed for implementation, these perception hypotheses are still acceptable in an actual robot application. They can be provided ideally either by an RGBD camera or by a fusion of camera + Lidar, thus adding the relative distance to the bearing information.

However, it should also be noted that the good results of Janosov and Hüttenrauch are based on hypotheses that are difficult to put into practice in real case pursuit, where the target is not cooperative. Although both strategies use relative distance as input (the first in polar coordinates and the second in Cartesian), both also consider observing the target's velocity in an inertial frame, which is an improbable assumption considering the current embedded sensors.

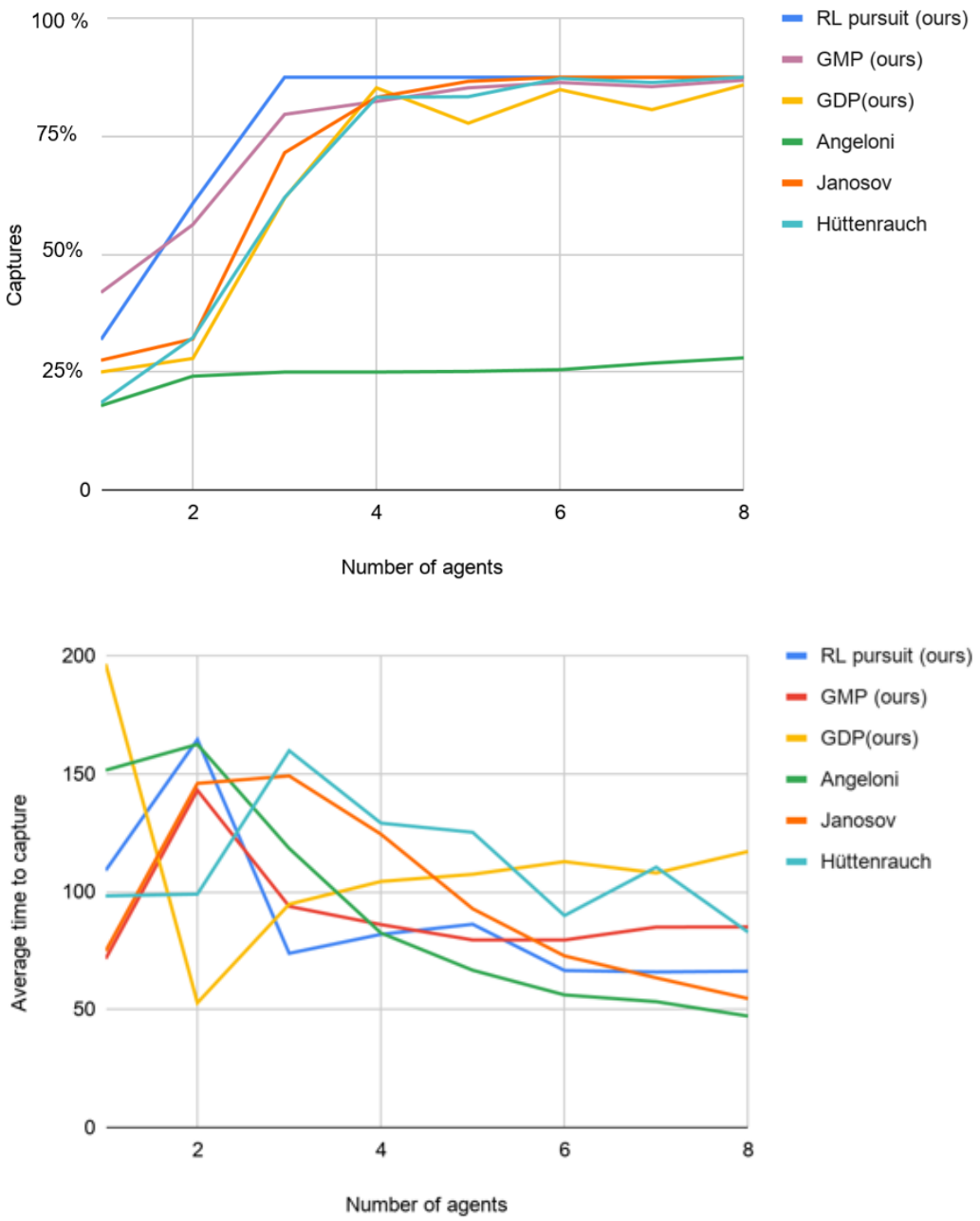


Figure 2.21 – Comparison between our proposed approaches and the baseline strategies.

2.5.3 Effects of the relative velocity

In the following two graphs, we will see the relative velocity (V_T/V_p) effect on the capture quantities. The results still maintain the same pattern presented in the previous experiments—our RL pursuit with the best yields, followed by GMP, Janosov, Huttenrauch, GDP, and Angelani. Besides, as expected, increasing the relative speed reduces the capture rate of all analyzed algorithms.

Besides, it is evident the Angelani's low performance for faster target ($V_T/V_p > 1$). It is predictable considering that they do not employ any "prediction" strategy for the target's pursuit. This implementation basically consists of a Pure Pursuit towards the target and the repulsion force between pursuers to avoid collisions.

The need for "prediction" for capturing faster targets had already been indicated in Janosov's work. However, their solutions rely on knowing the target's speed in global coordinates, which is a hard hypothesis considering a non-cooperative target. However, the GMP shows that good performance can be achieved when considering parallel navigation. As argued in state of the art, this phenomenon has optimal properties under certain pursuit conditions.

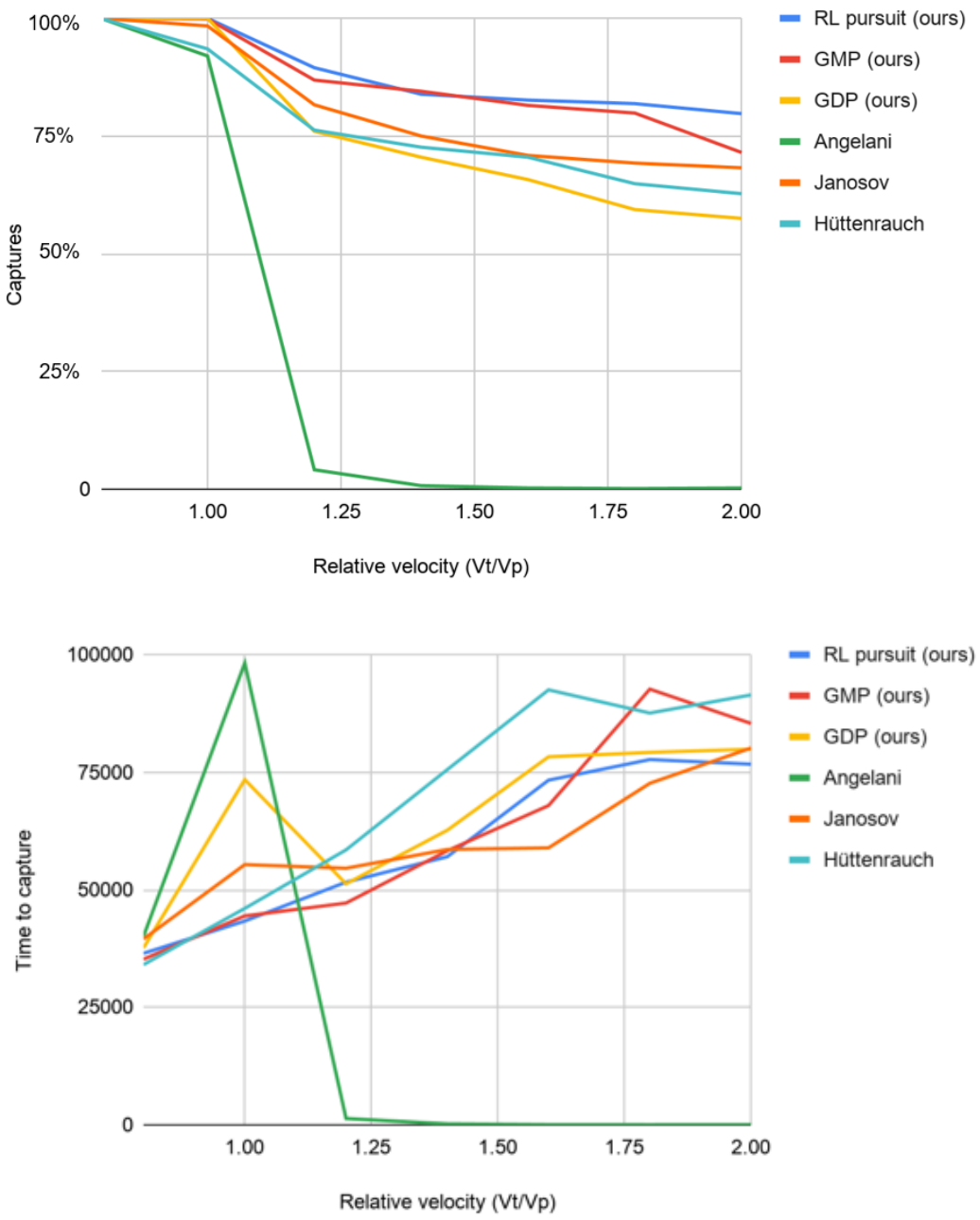


Figure 2.22 – Comparison between our proposed approaches and the baseline strategies.

2.5.4 Effects of the arena size

As we showed previously, with an increasing number of agents, the task decreases in difficulty for a fixed arena size. Therefore, with a larger number of agents, most approaches can perform the task successfully. The pursuit-evasion game is played in an obstacle-free arena, but the agents can use the arena boundaries to constrain the evader movements. In this section, we investigate the effect of larger arena sizes using n = 3 agents.

In the results exposed in Figure 2.23, the only parameter changed between the evaluations and the arena's radius size, which is multiplied by a factor (Radius factor). In the graphs below, we have the performance metrics depending on the arena's size, which varies between its standard size (Radius factor = 1) up to twice this (Radius factor = 2). Note that, although the loss of performance is common to all strategies, our strategy RL pursuit and GMP are less likely to suffer from the increase in the arena's size.

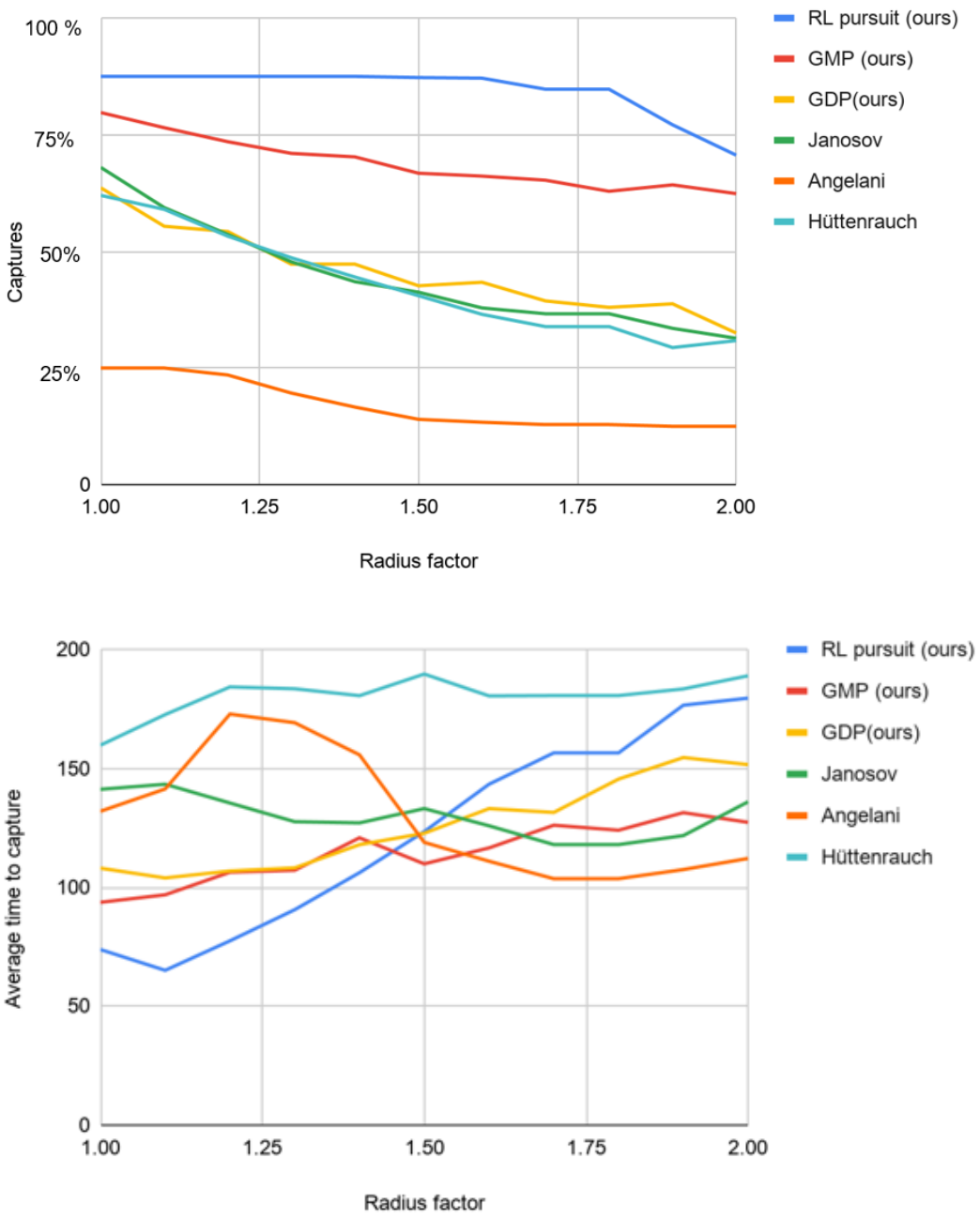


Figure 2.23 – Comparison between our proposed approaches and the baseline strategies.

2.6 Conclusion

In this chapter, we propose three different techniques to model predators' behavior in a planar pursuit. We also consider the case where the prey is faster than the pursuer, which requires an "intelligent" strategy from the pursuers. We also consider hypotheses consistent with real-world applications, such as non-holonomic motion and instantaneous observations in relative and polar coordinates. In none of the three cases of communication are necessary.

In the first approach, **Group Deviated Pursuit**, we propose a simple model using a "guidance law" to describe hunting behavior. We propose a modified version of the classic Deviated Pure Pursuit (DPP), where the offset angle is responsible for the different trajectories during an ambush. The offset is calculated instantly based on the relative engagement between pursuers and targets.

In the second proposition, **Group Mixed Pursuit**, we explore the use of another "guidance law," which is still very popular in missiles, the Proportional Navigation Guidance (PNG). Furthermore, its geometric rule, the "Parallel Navigation," is found in several natural predators' behavior. The PNG term provides a more efficient pursuit, while a DPP term is still maintained to avoid collisions and generate different trajectories.

Finally, in the third section, **Group Pursuit using RL**, we studied Deep Reinforcement Learning for the multi-agents Pursuit. In this approach, the hunting behavior is given Multi-Layer Neuronal Networks, in which the weighs were obtained by try-and-error in a simulated pursuit-evasion environment.

A qualitative analysis of this and other techniques will be present in the last chapter, **Discussion and Conclusion**, all the propositions will be submitted to a Benchmark, and different metrics will be highlighted.

Chapter 3

Improving algorithms: prediction, avoidance and flocking

In this chapter, we propose extensions for the previous group pursuit on the plane. We do not propose new models of pursuit behavior but complementary scenarios with different problem statements. In short, three new scenarios will be studied. In the first scenario, we will assume that the target's global position and velocity are available for the pursuers. With that, a predictor scheme is chosen to foresee the target's future position. In the second scenario, just one pursuer realizes the target's stalking; nevertheless, other non-cooperative agents, or mobile obstacles, are traveling in the same workspace. In a third and last scenario, we are interested in more massive amounts of agents, where not all of them are informed about the target. In this, we investigate the flocking formation considering only the relative and polar information about the neighbors.

Contents

3.1 Pursuit with target prediction

In this chapter, we assume that all pursuers have target information within a global framework. Based on this information and based on a predictive scheme, the target's future trajectory is estimated. Henceforth, the whole of pursuit is based on this collective knowledge of prediction and its uncertainty intervals.

This work is a cooperation between the Laboratory Heudiasyc - UTC and the Universidad Politecnica de Valencia. The results are in preparation for a close submission. The paper will be entitled: "**Autonomous drone pursuit using a fleet of drones with target prediction**" by C. de Souza Jr, A. Castillo, P. Castillo, B. Vidolov.

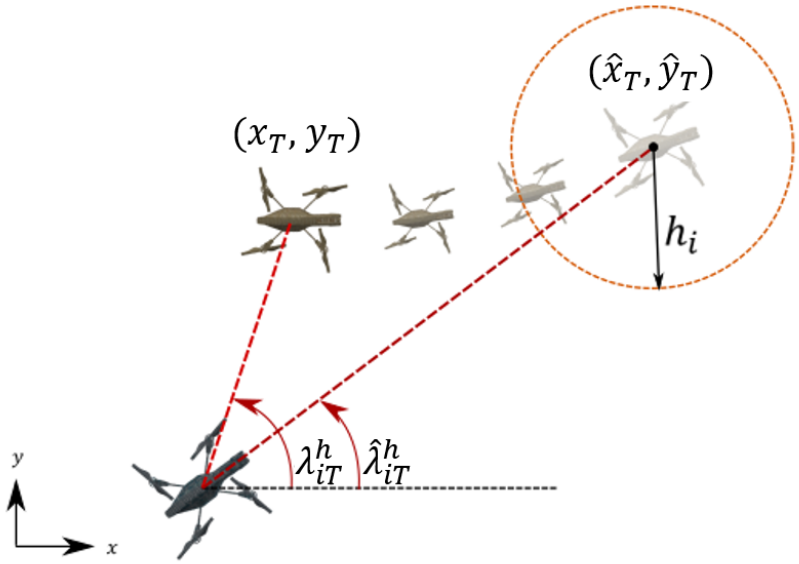


Figure 3.1 – Illustration of a pursuit with prediction. The pursuer (darker quadcopter) is pointing to the future position of the target $(\hat{x}_T, \hat{y}_T)$ instead its current position (x_T, y_T).

3.1.1 Brief preface and literature review

As seen briefly in Chapter 1, some of the anti-drone drones, such as [84, 134, 165], depend not only on the perception of the ADD itself but also on external location systems. These external locators can accurately deliver the target's location to pursuers and possibly improve the capture task's performance.

With the recent increase in illicit use of mini-drones, numerous studies have been dedicated to the problem of location. In [57] and in [166], the authors point out numerous approaches relevant for their mini-drone detection and tracking. Some of the most common solutions are audio detection, thermal detection, radar detection, and RF (radio frequency). Since these attacks require a quick and precise counter-response, state-of-the-art solutions are usually based on multi-sensors, such as *ADS-ZJU* and *DroneShield* [166], or smart-sensors, such as the *anti-UAV Defense System (AUDS)* [57].

In [165], the authors already considered the problem of an intruder's pursuit depending on an external target location system. They considered a system of beacons installed in an area, and the target positioning was obtained through triangulation techniques. Despite the similarities, this work focused more on location and less on the pursuit itself. Besides, this work considers a single-pursuer configuration, and its disadvantages have already been seen in State-of-the-art.

In [84], the authors consider the target's location made by a 3D air defense radar, and the information is transmitted to the pursuers. Although this work considers a group of drones, they are physically connected by a towed net. Consequently, as seen in Chapter 1, in the case of fixed formation, the multi-agent system behaves as a single body, losing the advantages of smart group strategies, such as ambushing and stalking.

In this work, we assume the deployment of a location system, in such way that the target position and velocity are delivered to the pursuers without compromising on delays. The pursuit rule is similar to the *Deviated Group Pursuit*, proposed in 2, where the offset angle provides the desired formation and collision avoidance for the group.

However, to enhance the capturability, a predictor scheme for estimating the target's future position is chosen, and the ambushing formation is based on the prediction error.

3.1.2 Problem formulations

Similar to Chapter 2, we also have a multi-agents pursuit-evasion problem involving N pursuers and a single evader moving in a horizontal plane. The conditions of capture and the kinematic model of pursuer (see equation 2.1) and target still are the same.

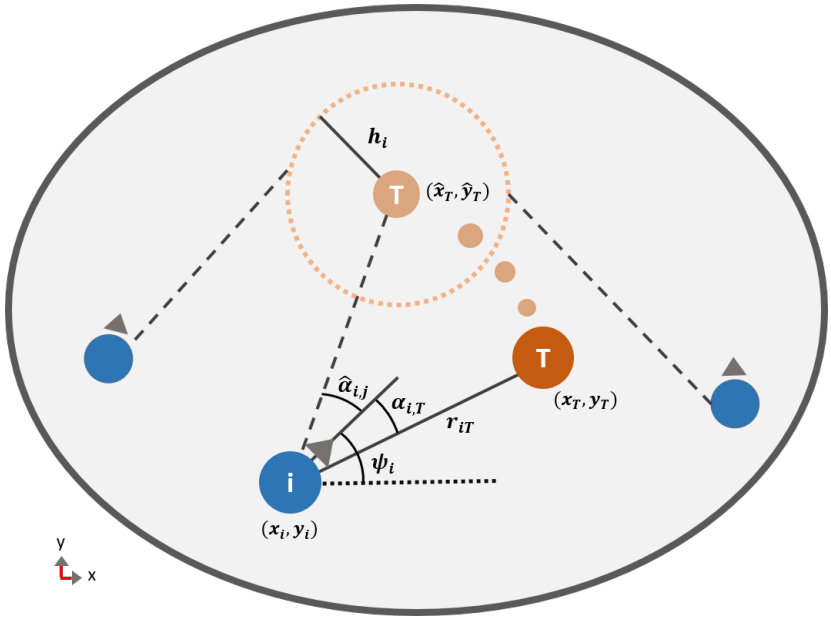


Figure 3.2 – Problem formulation of the predicted pursuit. All the pursuer knows the target position and velocity in the global frame and its foresee trajectory. Based on evader capabilities, the prediction error can be estimated h_i.

Nevertheless, while in the previous chapter, the target perception was done only by relative and polar coordinates, in this section, we suppose the target's position and velocity available in the global frame for all pursuers. In a practical scenario, the global information could be acquired either by beacons localization systems, such as in [165]; or by 3D radar systems, such as in [84]; or even by any technology of localization described in [57] and [166].

Furthermore, in this section, the pursuit will not be towards the instantaneous position of the target, but for a predicted future position $(\hat{x_T}, \hat{y_T})$. Details about the predictor scheme and estimation of the future position are given next.

3.1.3 Target prediction

The simple pursuit, towards the instantaneous target's position, frequently leads to a tail pursuit. Nevertheless, one natural behavior for a chaser should be to take advantage of the target's known states and by predictions to point towards its future position, reducing efforts as depicted in Figure 3.1.

In this work, a predictor scheme, initially proposed in [172], is used for estimating future target positions defined in a prediction horizon. For this, we consider that all pursuers have information about the current position of the target and other agents in the inertial frame.

Target position ahead

Denote by x_T (m) the target position, by v_T (m/s) the target velocity and by m_T (kg) the target mass. Then, using Newton-Euler, its movement is defined as

$$\begin{aligned} \dot{x}_T(t) &= v_T(t), \\ \dot{v}_T(t) &= \frac{1}{m_T} F(t) \triangleq a_{T_0}(t). \end{aligned} \tag{3.1}$$

where $F(t)$ defines the external force producing its movement. From (3.1), the future target-position, $x_T(t + \Delta_T)$, is

$$x_T(t + \Delta_T) = x_T(t) + \Delta_T v_T(t) + \int_T^{t+\Delta_T} (t + \Delta_T - s) a_{T_0}(s) ds. \tag{3.2}$$

Observe that equation (3.2) provides the desired value $x_T(t + \Delta_T)$, nevertheless it is not possible computed it directly because $a_{T_0}(s)$ is not known for $s \in [t, t + \Delta_T]$. However, consider that $a_{T_0}(t)$ and its first i derivatives can be estimated. Therefore, by Taylor, the function $a_{T_0}(s)$ could be reconstructed, for

all $s \in [t, t+\Delta_T]$, as

$$a_{T_0}(s) \approx \hat{a}_{T_0}(s) \triangleq a_{T_0}(t) + (s-t)\dot{a}_{T_0}(t) + ... + \frac{(s-t)^r}{r!} a^{(i)}_{T_0}(t), \tag{3.3}$$

with an error, $h_i(s) \triangleq a_{T_0}(s) - \hat{a}_{T_0}(s)$, bounded by

$$h_i \leq \frac{(\Delta_T)^{i+1}}{(i+1)!}\mu \tag{3.4}$$

being $\mu \triangleq \max_{\eta\in[t,s]}\{a^{(i+1)}_{T_0}(\eta)\}$.

From (3.2)-(3.4), the future target-position, $x_T(t+\Delta_T)$, can be estimated by

$$\hat{x}_T(t+\Delta_T) = x_T(t) + \Delta_T v_T(t) + \int_T^{t+\Delta_T} (t+\Delta_T - s)\hat{a}_{T_0}(s)ds. \tag{3.5}$$

with an error bounded by

$$x_T(t+\Delta_T) - \hat{x}_T(t+\Delta_T) \leq \frac{(\Delta_T)^{i+2}}{(i+1)!}\mu \tag{3.6}$$

Therefore, for estimating the future target position, the drone needs to compute equations (3.3), (3.5) in real-time. The variables a_{T_0}, $\dot{a}_{T_0}$, ..., $a^{(i)}_{T_0}$, are estimated by using

$$\begin{aligned}
\dot{\hat{x}}_t &= \hat{v}_T + l_{11}(x_T - \hat{x}_T) + l_{21}(v_T - \hat{v}_T),\\
\dot{\hat{v}}_t &= \hat{a}_{T_0} + l_{12}(x_T - \hat{x}_T) + l_{22}(v_T - \hat{v}_T),\\
\dot{\hat{a}}_{T_0} &= \hat{a}_{T_1} + l_{13}(x_T - \hat{x}_T) + l_{23}(v_T - \hat{v}_T),\\
&\vdots\\
\dot{\hat{a}}_{T_i} &= l_{1i}(x_T - \hat{x}_T) + l_{2i}(v_T - \hat{v}_T),
\end{aligned} \tag{3.7}$$

being $\hat{x}_T$, $\hat{v}_T$, estimate values of x_T, v_T, respectively; and $\hat{a}_{T_0}$, $\hat{a}_{T_1}$, $\hat{a}_{T_2}$, ..., $\hat{a}_{T_i}$ estimates of a_{T_0}, $\dot{a}_{T_0}$, $\ddot{a}_{T_0}$, ..., $a^{(i)}_{T_0}$, respectively.

Interval of prediction (Δ_T)

The Δ_T represents how far in time will be the target's prediction. For that we rely in the idea of time-to-interception, considering the current distance r_{iT} and assuming a constant closing rate $\dot{r}_{iT}$ (measured by the pursuer):

$$\Delta_T = \left[\frac{r_{iT}}{\dot{r}_{iT}}, T_{max}\right]_{min} \tag{3.8}$$

The up-bound (T_{max}) is determined experimentally in order to avoid outliers errors in the target prediction.

Once obtained Δ_T, the predicted position $\hat{\boldsymbol{x}_T} \triangleq \boldsymbol{x}_T(t + \Delta_T)$ can be determined and used in the pursuit. Consecutively, in the following sections the pursuit engagement will be described in relation to the predicted target $\hat{\boldsymbol{x}_T}$, as illustrated in Figure 3.1.

3.1.4 Pursuit with prediction

Similar to Chapter 2, let us considering the guidance law for implementing a deviated pursuit (DPP):

$$f_{\psi_i} = K_p * (\psi_i - \lambda_{iT} - \alpha_i) \tag{3.9}$$

where f_{ψ_i} is the control input for the heading velocity (ψ_i), and α_0 is the offset angle. Note that, considering (1.1), with the pursuer under (3.9), one necessary conditions for convergence of r_{iT} are $\alpha_i < \pi/2$.

The selection of different offsets α can results in different non-crossing trajectories, as can be seen in Figure 2.4. These curvilinear trajectories can be used as pursuer's behavior to ambush the prey from different directions. Furthermore, the paths described in Figure 2.4 can be easily compared to the description of lionesses chasing trajectories given by [45].

In order to create this ambushing behavior, let consider the following offset angle's function:

$$\alpha_i = \begin{cases} \hat{\alpha}_{iT_0} * \left(\frac{\sum_{j!=i}^{N} \delta_{ij}(\xi_i, \xi_j)}{N-1} \right) & N > 1 \\ 0, & N = 1 \end{cases}$$

where N is the number of neighbors of the i-agent and δ_{ij} is a summation function previously described in (2.3). In short, the outcome is similar to repulsive forces applied in the perpendicular direction of the LOS. Thus, δ_{ij} can be written as:

$$\delta_{ij} \begin{cases} -1, & \text{if } j \text{ is in the left side} \\ 1, & \text{if } j \text{ is in right the side} \end{cases}$$

The offset reference ($\hat{\alpha}_{iT_0}$), is the maximum offset angle that one agent can assume in a pursuit, and it is based in the prediction error (h_i), previously explained (eq. 3.4) and in the distance to target $|r_{iT}|$.

$$\hat{\alpha}_{iT_0} = \left[\alpha_0, \tan^{-1}\left(\frac{h_i}{r_{iT}} \right) \right]_{max} \tag{3.10}$$

where α_0 is the minimum offset kept to avoid collision between two neighbors. The principle behind this is that the pursuers in the extremities will travel towards the extremes case in the target prediction, i.e., they will assume the higher error in the prediction and intercept the target there.

3.1.5 Simulations

One pursuer vs one intruder

Figure 3.3 illustrates the performance when a pursuer tracks one intruder. The red circle is the predicted target position intersection, and the pink circle is the estimated error in the prediction. In this scenario, the target starts in the position $(x, y) = (-7, 3)$m and moves with constant velocity in its positive longitudinal axis. Observe that the estimation stabilizes and the pursuer is in the intersection target route after the third frame.

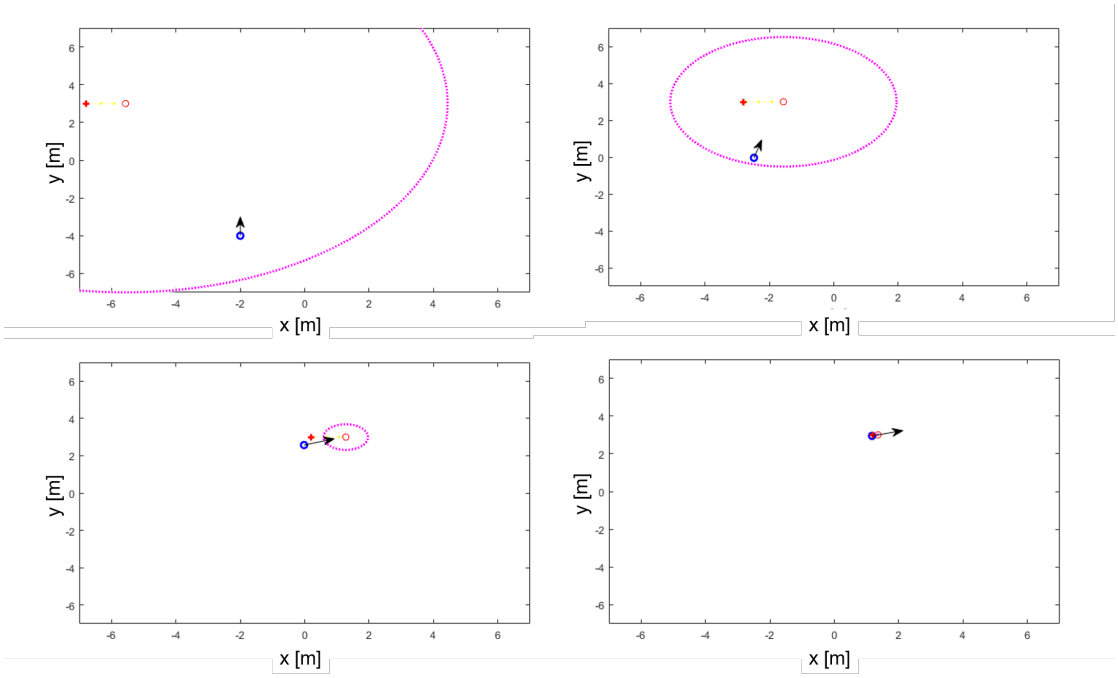


Figure 3.3 – Single pursuer against an evader. The pursuer (blue circle) heads towards the future position of the target (red circle) instead of its instantaneous position (red cross). The violet circle denotes the "error in the prediction".

Comparison with Group Deviated Pursuit

In Figure 3.4, two scenarios with three pursuers' pursuit are illustrated; one using the predictor scheme, Figure 3.4-top, and the other using the simple GDP, Figure 3.4-bottom. Similar to the previous case, the target assumes a constant positive velocity on its x-axis. Observe that when using the prediction scheme, the performance is improved, and the total trajectory traveled for each pursuer is smaller than for the first case.

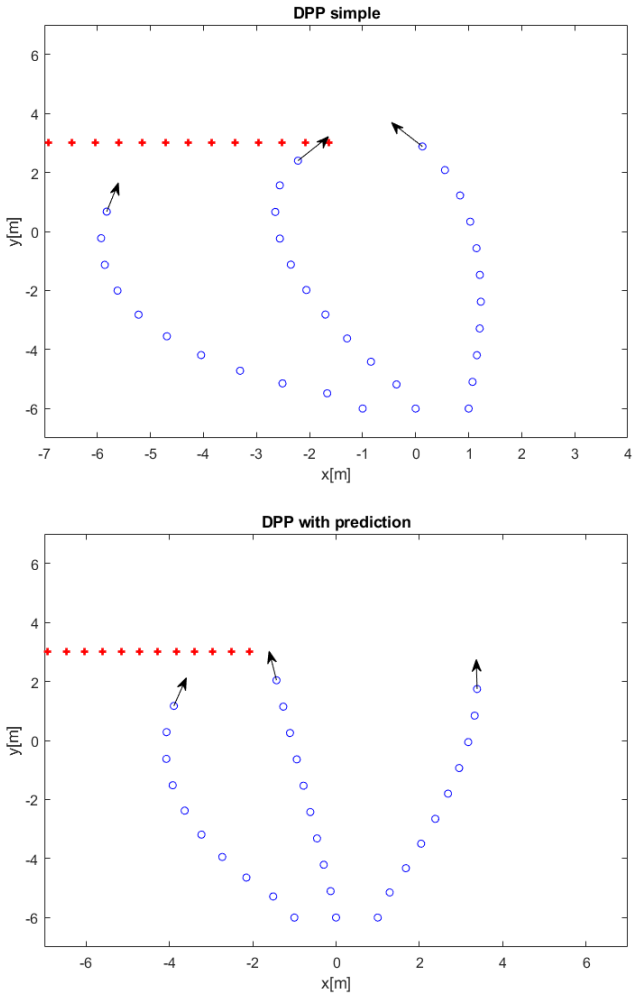


Figure 3.4 – Pursuers behaviors when hunting an intruder drone using the DPP based methodology. *Top*: without prediction algorithm and *bottom*- with the predictor scheme.

3.2 Pursuit with non-cooperative agents

This section will address a different problem in multi-agents pursuit, where a single pursuer must track a target in an environment with non-cooperative agents. From the pursuer point of view, these non-cooperative agents can be seen as mobile obstacles, and then the problem becomes similar to the traditional obstacle evasion. In this strategy, instead of using conventional repulsive force to avoid the obstacles or find a route through an optimization problem, we propose a simple and reactive solution based on the "straight-line condition course." This principle is the same for the traditional family of navigation guidance PNG (Proportional Navigation Guidance).

This work was published in the 23rd IEEE International Conference on Intelligent Transportation System, under the title of **Reactive drone pursuit and obstacle avoidance based on parallel navigation**, in September 2020.

Figure 3.5 – Illustration of a futuristic city, where the autonomous drone would tracking a target in a crowded and shared air-space.

3.2.1 Brief preface and literature review

Tracking a mobile object in real time is a demanding task in the UAV field; it can be used in several applications, such as surveillance, cinema recordings, and sports broadcasting. Although several commercial drones already feature tracking the desired object autonomously using its on-board sensors, collision avoidance with mobile obstacles is still a drawback for its applications in more

complex scenarios, such as navigation in a big city. Besides, drones' futures applications will require multiple drones sharing the same aerial space while attending different tasks. Those non-cooperative agents must make fast decisions to avoid each other, as vehicles and pedestrians cross each other in a big city.

As reviewed in the Chapter 1, numerous examples of robotic applications have used GL for tracking. Moreover, its uses are the most varied, from autonomous navigation[17, 1], passing through autonomous landing [41], until interception of intruder drones by ADDs [3, 28]. However, unlike its original application, missile guidance, mobile robots' working area constantly also has the presence of obstacles, which can be mobile or static. However, for an effective GL in mobile robotics applications, obstacle avoidance techniques must also be incorporated into the system.

Many efforts have been made to conciliate the guidance laws to the opposite case, collision avoidance. One recurrent solution adds the use of changing modes switching between pursuit and avoidance task. In [20] and [17], the authors considered an intermediate goal, a collision-free point, for the pursuer to go until the overcoming of the obstacle. One alternative is the use of the parallel navigation principle in the collision avoidance [18], [15], and [23]. In [18] the authors proposed a PNG based method Line of Sight Counteraction Navigation (LOSCAN) to avoid collision between ships; the algorithm proposes command to be executed by the sailor in order two avoid a collision.

Similarly, in [23], the author introduces the concept of a virtual plane to identify the collision based on the rendezvous-course condition. In this plane, the dynamic moving obstacles are transformed into stationary objects, and through it, the navigable paths are obtained. In [15] is proposed collision avoidance guidance laws based on the PNG principles to sense and avoid capabilities in a UAV, their solution for multiples target is based on objective-based cost functions to find the optimal path.

In this work, we propose a navigation guidance-based solution for tracking and avoiding non-cooperative agents or obstacles. We extended the traditional Proportional Navigation Guidance (PNG) for collision avoidance of multiples obstacles, including mobile objects with random movements. This work differs from the previous in the following aspects:

no need of changing mode between pursuit and avoidance, as in [20], [17], in our strategy, the pursuit and evasion are components of the same guidance law;

does not rely on the optimization calculation as in [15]; being a computationally fast solution and easy to implement.

A comparative of the proposed algorithm with respect to the pure pursuit is presented with numerical simulations. Different scenarios are provided for verifying the good performance of the proposed algorithm.

3.2.2 Problem statement

The challenge is to propose a solution for a mobile target's drone tracking problem in a dynamic environment where non-cooperative agents are evolving. This signifies that the pursuer drone must reduce its distance to the target (r_{iT}) and avoid the collision with other agents (r_{ij}).

The scenario is illustrated in Figure 3.6, the pursuer (blue circle) must track the target (red circle) while avoiding collision with the non-cooperative agents (grey circles) flying with unknown movements. The green zone in the picture illustrates the perception zone where obstacles outside will be ignored. All agents in this work have equal capacities and the same configuration. Also, the pursuer will have a velocity equal to or higher than the target and obstacles.

3.2.3 Pursuit and avoidance guidance law

In order to determine the steering law for f_{ψ_i}, let us first consider the PNG (Proportional Navigation Guidance), previously exposed in Chapter 2. This basics principle is to keep a constant bearing angle towards the target. Moreover,

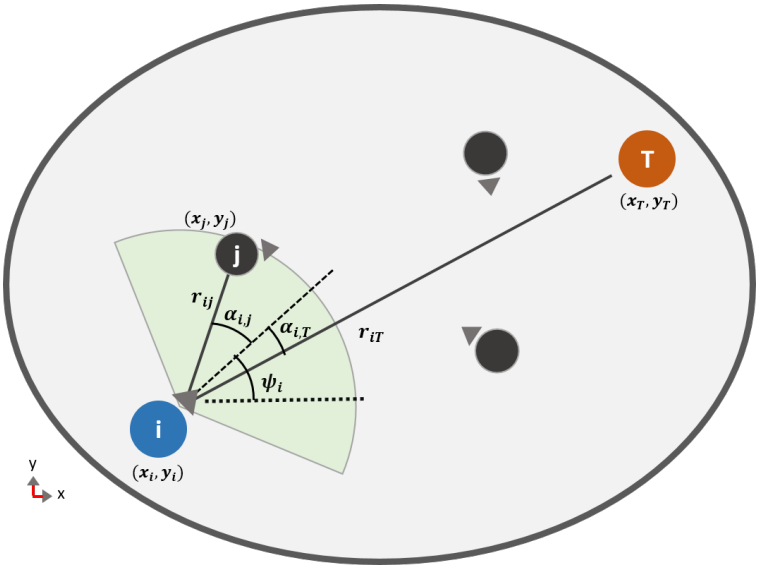


Figure 3.6 – Engagement between the pursuer (blue circle), the target (red circle) and one non-cooperative agent j (grey circles). The green area represents the obstacle's perception area.

a logical choice for a controller to nullify $\dot{\lambda}_{iT}$ can be defined as:

$$f_{\psi_i} = N * \dot{\lambda}_{iT} \tag{3.11}$$

where N represents the gain navigation constant and $\dot{\lambda}_{iT}$ is the angular rate of the bearing angle between pursuer and target. Remark that a constant bearing is a necessary condition for the collision as stated in (3.1), hence, for assuring the inequality of this condition, it must be enough to assure the non-convergence between two bodies, i.e., in a navigation scenario, the avoidance.

Therefore, the function $f(x) = x^{-1}$ is chosen for assuring the inequality $\dot{\lambda}_{ij} \neq 0$, and the following avoidance component for a guidance law can be proposed:

$$f_{\psi_i} = \frac{-k}{\dot{\lambda}_{ij}} \tag{3.12}$$

where λ_{ij} is the angular rate of bearing angle and and k denotes a positive gain. The negative sign assures that the pursuer heading in the opposite direction of λ_{ij}.

Although the above equation can assure the non-collision with a single obstacle, it can be an inconvenient due the fact the distance is not taken into account, and in a multi-obstacles scenario a closer obstacle will not have priority over a further one. For overcome this issue a second term is proposed, where the control output will be weighted based in the inverse of its relative distance (d_{ij}). Thus

$$f_{\psi_i} = \frac{-k}{\dot{\lambda}_{ij}} * \sigma\left(\frac{R_{col} - r_{ij}}{R_{per} - R_{col}}\right) \tag{3.13}$$

For this new term, R_{col} and R_{per} stand for the perception and collision radius, respectively. $\sigma()$ function can be described as:

$$\sigma(x) = \begin{cases} 0, & \text{if } x > R_{per} \\ x, & \text{if } R_{per} \geq x \geq R_{col} \\ 1, & \text{if } x < R_{col} \end{cases} \tag{3.14}$$

Observe from (3.13) the second term in the equation expresses a kind of collision coefficient which has its maximum value for distances smaller than R_{col} and null value if the distance is higher than R_{per}. Besides, this function increases the repulsion force for closer objects or to ignore others that are not in the eminence of collision.

Finally, the complete equation for target tracking and obstacle avoidance is proposed as

$$f_{\psi_i} = N * \dot{\lambda}_{iT} + \frac{-k}{\dot{\lambda}_{ij}} * \sum_{j \in O} * \sigma\left(\frac{R_{col} - r_{ij}}{R_{per} - R_{col}}\right) \tag{3.15}$$

where O defines the set of all non-cooperative agents (or obstacles) inside the perception area, see Figure 3.6. This equation combines the PNG (3.11) and the avoidance (3.13) generalized for the multi-obstacle case.

3.2.4 Simulations

To validate numerically our proposition, we implemented the proposed solution in Matlab, considering the agents modeled as a particle moving in a two-dimensional space and under the kinematics equations (3.16).

The sample period, T_e in simulations is $0:01$s. For simulation purposes, the crosses represent the evader's path, while the circles illustrate the pursuers' path. The black arrows denote the heading orientation of each pursuer. The collision condition in simulation is the target be in a distance less than the capture radius $R_{min} = 0.3$ m of one agent.

Fixed obstacles, fixed target In the following scenario, the pursuer must navigate between numerous static targets using the pursuit-avoidance strategy (3.15). The trajectories can be seen in Figure 3.7.

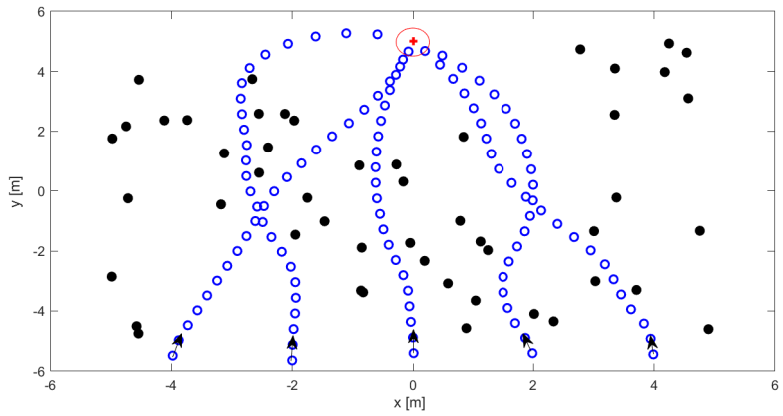


Figure 3.7 – A pursuer autonomously navigating into an environment with multiples fixed obstacles for tracking a static target.

In this figure, several episodes were taken considering different initial positions for the pursuers. Notice that the pursuer does not know the global location of all obstacles; therefore, the optimal path can not be assured. However, remark that this approach provides a smooth collision-free path towards the target using a simple and fast calculation algorithm.

Fixed obstacles, moving target A slightly different scenario is prosed here. Random obstacles were also placed, but now the target is executing a linear trajectory; see Figure 3.8. Observe in this figure the successful target capture and the well-defined obstacle avoidance.

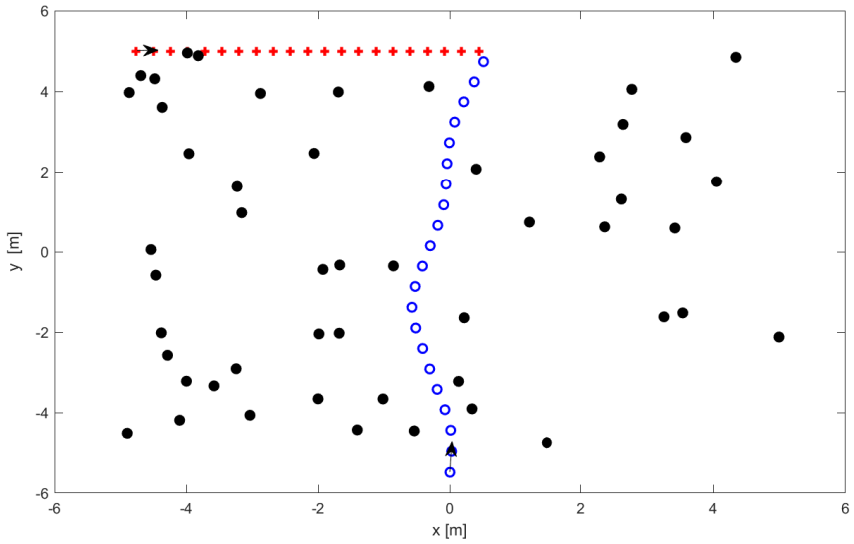


Figure 3.8 – A pursuer autonomous navigating into an environment with multiples fixed obstacles for tracking a moving target.

Moving obstacles: In this last scenario, the pursuer must track the target into an environment where non-cooperative agents are evolving. These non-cooperative agents are moving with different velocities along the way of the pursuer towards the target, as can be seen in Figure 3.9. For this simulation, the agents only move in the x-axis; nevertheless, the result can be easily extended for non-cooperative agents moving in 3D.

In this scenario, the pursuer starts at position $(x_0, y_0) = (0, -6)$m and displaces successfully between the non-cooperative agents toward the target positioning at $(x_T, y_T) = (0, 5)$m. Observe, still from Figure 3.9, the good performance of the algorithm facing a complex scenario with multiples accelerating non-cooperative agents.

Observe that the principle is not a pair-wise repulsion force as a function of the distance from the pursuer to the obstacle. Instead, to avoid the nullification of bearing-angle rate pursuer-obstacle ($\dot{\lambda}_{iT}$), consecutive, even if an obstacle is close to the pursuer, it will not change its course if they are not in collision route.

For example, from the screenshots 4 in Figure 3.9, we can remark easily that even with the last obstacle passing close to the pursuer, i.e., inside the perception radio (R_{per}), it does not affect the route of the pursuer, once they were not in collision route.

Note also, the pursuer observes only relative and polar based information ($\dot{\lambda}_{iT}$,d_{iT}) from the obstacles, which can be easier obtained from embedded sensors, such as camera and lidar.

The simulation results can be observed in the following link:
`https://youtu.be/ctrN42mLIUQ`.

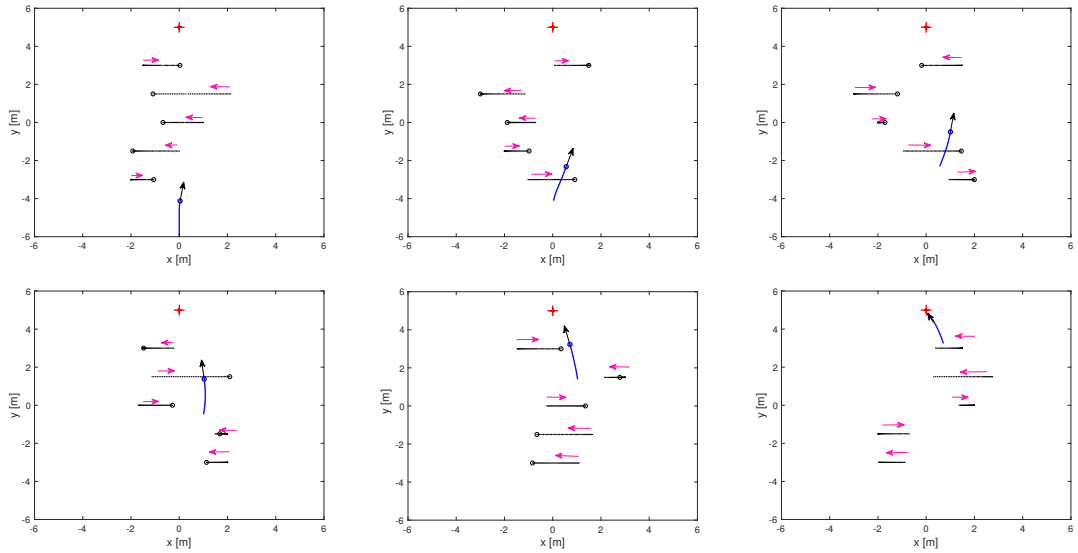


Figure 3.9 – Pursuer performance when navigating into an environment with non-cooperative agents are evolving.

3.3 Flocking

This last section will tackle a slightly different problem from the previous ones; we will not study group-pursuit strategies but the collective herd movement. However, we will see that the flock's principle is very similar to the used in group-pursuit, which will lead us to propose a generalized guidance law, addressing at once the pursuit, obstacle avoidance, and flocking. Our flocking model is based on the principle of "parallel navigation," where the flock's individual tends to maintain a constant bearing angle about the estimated flock's centroid. Moreover, we show that a cohesive motion can be obtained with only relative and polar neighbors' information.

This section's content is part of an article, still in the process of preparation, which will be entitled: **Flocking, pursuit, and avoidance: a constant bearing guidance law for multi-agents.** C. de Souza, P. Castillo, B. Vidolov.

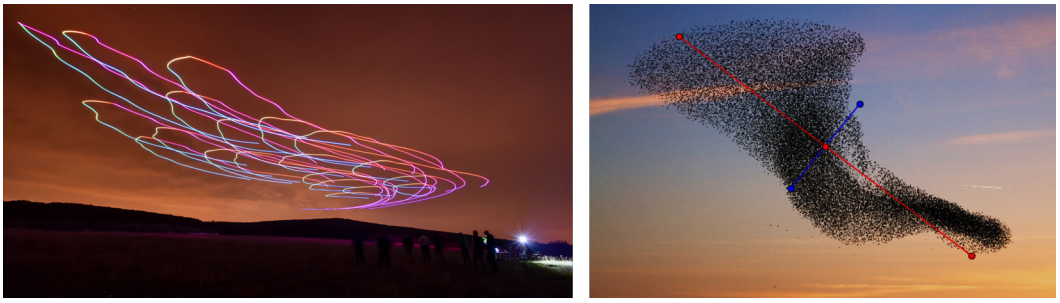


Figure 3.10 – Examples of flocking: on the left, a flock of autonomous drones from [148]. On the Right, Extract from [111], an example of centroid's identification from a bird's swarm image.

3.3.1 Brief preface and literature review

Flocking is not an exclusive behavior of herd's and school's animals, such as sheep, birds, and fishes. Far from that, Vicsek and Zafeiris' survey [50] points out that collective, coordinated, and cohesive movements are present in practically all moving beings, including physical particles, bacteria, insects, or even humans.

Considering this, we propose a flocking model that could be compatible with the previously proposed group-pursuit models. In this way, the pursuers, even not being in a pursuit action, could maintain a cohesive collective movement, and the transition between flocking to the pursuit, and vice-versa, could be done smoothly.

Before going any further, it is important to remind the distinction assumed in this thesis about "Flocking" and other collective movements. As pointed out in the introductory of this thesis, we refer by flocking "the fluid and cohesive motion promoted by agent-based and decentralized approaches, with homogeneous individuals, having no explicitly hierarchy and able to sense and act only locally." Differentiating so from all centralized or pattern formation approaches.

As exposed in the Chapter 1, there are numerous models for flocking in the literature. They range from the simplest ones, such as primary Vicsek particle [77], passing through deterministic models, such as Cucker–Smale [138], and its numerous variation and control-theoretical frame-work, such as in [58]. Furthermore, several successful attempts have implemented into drones, such as [53, 54, 80, 148], where several dozen quadcopters can display amazing formations in an opened sky, see Figure 3.10.

Nevertheless, the above implementation relies on communication between neighbors to share their position and velocities in the global frame. This assumption seems to be not very verisimilar compared to the observed real-life flock. This phenomenon is observed even in the most simplistic life, where no communication is identified. One realistic flocking should be obtained, relying only on relative information that could be easily obtained from embedded sensors, such as cameras or lidars.

Furthermore, the problem is those classic algorithms, in addition to the components of attraction and repulsion (attractive/repulsive pairwise potential forces), also require either term of heading's alignment, such as [49, 168], or velocity consensus (viscosity term), such as in [58, 77, 138]. These terms guarantee cohesion in the movement since the individual's velocity or orientation is decided based on the average of those measures of his neighbors.

However, measuring the orientation, or the absolute speed of another agent, is not an easy task. What makes a large part of flocking work is to assume a communication between agents, sharing this information among themselves.

Therefore, we propose a solution where neighbors' velocity, or orientation, does not need to be measured. We rely on the *centroid* estimation of the perceived neighbors, and we apply "parallel navigation" towards it. The Flock Centroid FC is a feature that can be obtained by image processing since it can be stated as a simple optimization problem, such as exposed in [111]. In short, our approach proposes that velocity alignment is not a requirement for flock formation. Instead, alignment is the outcome when applying simple geometrical rules that will be described next.

3.3.2 Problem statement

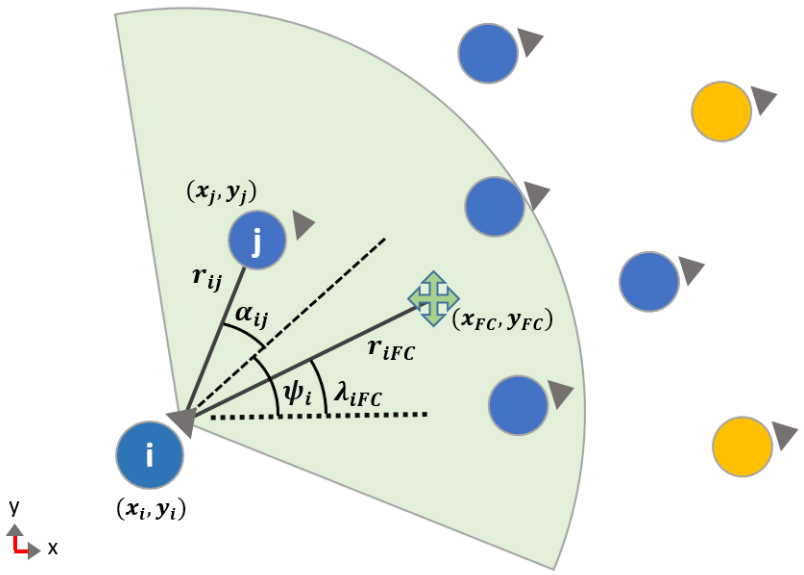


Figure 3.11 – Geometry between one agent i and the rest of the flock. Its closest neighbors are identified as j, and the flock's centroid FC is denoted by a green cross. Note that the FC is estimated based on the neighbors inside the field of vision (green area). The yellow circles represent the "informed agents."

Let us consider the flocking problem involving N agents moving in a horizontal plane, such as illustrated in Figure 3.11. The agent's path is described by $P_i(t) = [x_i(t), y_i(t)]$, where (x_i, y_i) is its Cartesian coordinates in the bounded work-space $W \in \mathbb{R}^2$. The initial goal for each agent is to keep the group's cohesion while do not collide with each other. The agents are evolving under the non-holonomic equations (3.16) and their behavior is controlled by the dual input $[f_{\psi_i}, f_{v_i}]$, which are the angular rate and the linear velocity, respectively.

An agent perceives its environment in polar and relative coordinates, i.e., the distance r_{ix} and a bearing angle λ_{ix}, between the agent i and the object x. Polar and relative coordinates seem a coherent assumption since that information can be obtained from a common robot's embedded sensors, such as a camera and lidar. Besides, we suppose that agents can infer their heading (ψ_i) in the global frame, which is also a consistent assumption for robotic applications, considering electronic compass sensors.

To adopt more realistic conditions we also suppose a limited fiel-of-vision FoV, characterized by an opening angle β and a radius R_F. The agent i is able to perceive only the neighbors inside FoV_i. Finally, an individual must also be able to estimate its perceived flocking centroid FC^i, which is the center of mass of all agents inside the field-of-vision FoV_i, i.e., $FC^i = \frac{1}{N_F} \sum_{j \in FoV_i} P_j$, as illustrated in Figure 3.11.

Finally, to realize more complex collective behaviors, we borrow the concepts of "informed individuals" from [169], which consist of specific individuals who have the flock's mission. In a certain sense, they can be considered "leaders," but we kept its original homogeneity concepts, where one agent can not distinguish from whom is informed or not.

3.3.3 Flocking model

Simplistic model

In this flocking strategy, the agent i regulates its orientation to keep a constant bearing angle to respect to FC^i, i.e., $\dot{\lambda}_{FC^i} = 0$. For its turn, the linear velocity of i is regulated to keep a safety distance toward the closest agent. The simplest form for this flocking is stated below:

$$f_{\psi_i} = K_f * \dot{\lambda}_{FC^i} \tag{3.16a}$$

$$f_{v_i} = u_{max} * \sigma_a \left(\frac{r_{min} - R_{saf}}{R_{per} - R_{saf}} \right) \tag{3.16b}$$

where K_f defines a positive gain and $\dot{\lambda}_{FC^i}$ is the bearing angle rate between pursuer and its observed centroid (FC^i). Also, R_{saf} denotes the closest safety distance between two agents, and R_{per} is the perception radius, which is the limit to how far the neighbor an agent will be perceived by another. This function makes the pursuer start to break from R_{per} linearly until to achieve R_{saf}. The $\sigma_a(x)$ is a function defined in $x \in IR \mid x \geq 0$:

$$\sigma(x) = \begin{cases} 0, & \text{if } x > R_{per} \\ x, & \text{if } R_{per} \geq x \geq R_{saf} \end{cases} \tag{3.17}$$

Note that at this point, each agent tends toward the perceived center of mass, since under the PNG, if $v_i > v_{FC^i}$ then $\dot{r}_{FC^i} < 0$.

Alignment In order to illustrate how the alignment of heading can be achieved using the law (3.16), let us consider two agents, a target T, and a follower i. The target, which can be seen as the informed agent, is executing a non-maneuvering trajectory with v_T positive and constant. The follower i, is operating under the flocking law (3.16), and since T is the only member of the flocking, $\lambda_{FC^i} = \lambda_{iT}$. Now, considering the relative kinematic equations described in (1.1), and considering that under (3.16), $\dot{\lambda}_{iT} \to 0$ and $\dot{r}_{iT} \to 0$, thus, the system (1.1) becomes:

$$\begin{aligned} v_i * \cos(\alpha_i) &= v_T * \cos(\alpha_T), \\ v_i * \sin(\alpha_i) &= v_T * \sin(\alpha_T). \end{aligned} \tag{3.18}$$

where the clear solution are $v_i = v_T$ and $\alpha_i = \alpha_T$, which indicates the alignment, i.e., same linear velocity and same heading for i and T.

Generalization: pursuit, avoidance and flocking

We proposed a unified guidance law responsible for the behavior of tracking, flocking and collision avoidance. The steering control input is given by the sum of the components:

$$f_{\psi_i} = f_{\psi_i}^{track} + f_{\psi_i}^{avoid} + f_{\psi_i}^{flock} \tag{3.19}$$

The track component $f_{\psi_i}^{track}$ can be given by (2.9) or by other guidance law, such as in Sections 3 or 4. The avoidance component $f_{\psi_i}^{avoid}$ is stated in (3.12), and finally, the flocking component $f_{\psi_i}^{flock}$ can be given by (3.16).

Leadership strategies can be established to conciliate the conflict between "flocking" and "tracking" tendencies. For example, just the closest agents can see the target, while others attempt to maintain cohesion with the group. Alternatively, heterogeneous weights can be given for the agents. Making in some agents a stronger track impulse($f_{\psi_i}^{track}$)over the cohesion impulse ($f_{\psi_i}^{flock}$), thus making predetermined leaders.

3.3.4 Simulations

Next, three scenarios are proposed to demonstrate the capabilities of our proposition. The simulations were done through Matlab, and the agents are implemented by using the kinematic equation (3.16).

Consensus achievement In this scenario, 50 agents are placed randomly inside a square arena of 100x100. The agents have their velocity and turning rate controlled by the flocking law (3.16). In this first example, there are no informed

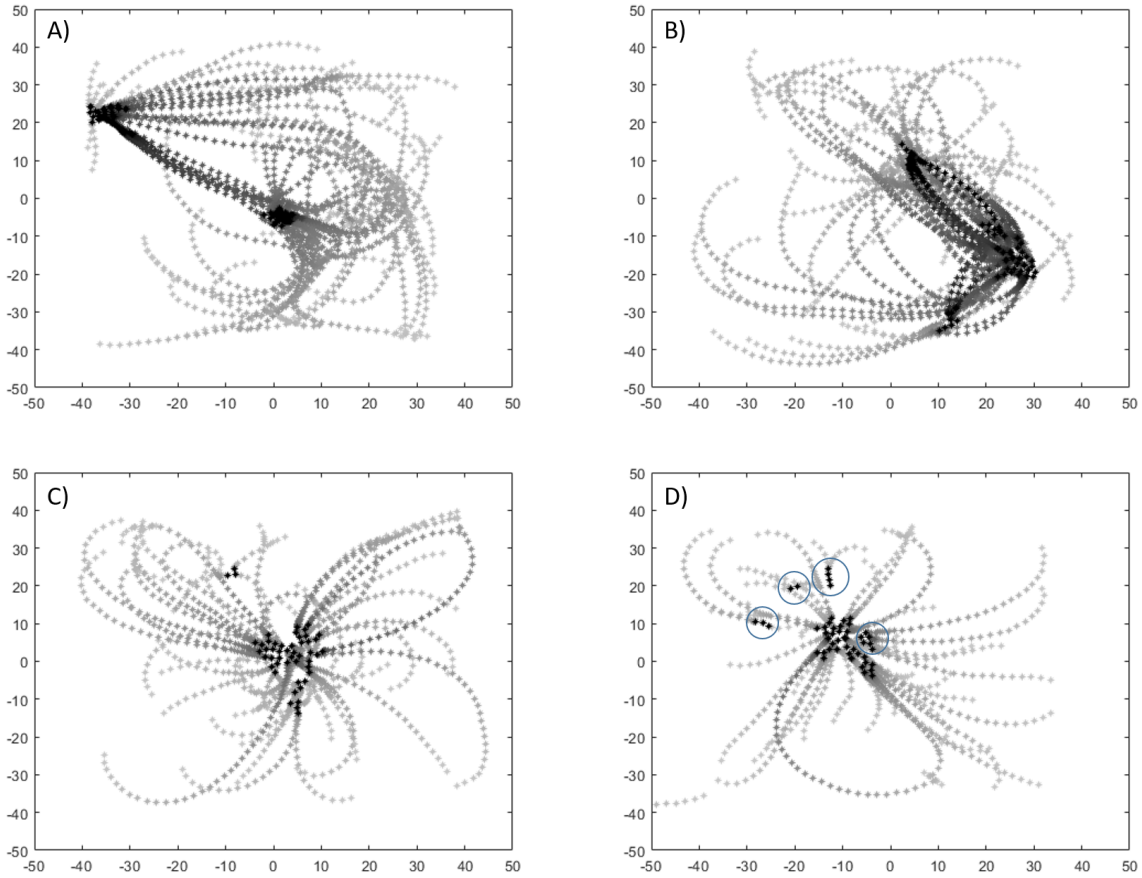


Figure 3.12 – Paths for aggregation under different opening angle (β): $\beta = \pi/4$ in A, $\beta = \pi/2$ in B, $\beta = \pi$ in C and $\beta = 2 * \pi$ in D.

agents, and all agents are equally tuned. In order to show the effect of the field-of-view, we proposed three episodes for three values of opening angle $\beta = \pi/4, \pi/2, \pi, 2 * \pi$, while the FoV radius keeps the same, $R_F = 30$.

In Figure 3.12, note that for smaller values of the field-of-vision (A and B), the transitory phase is longer, and we can see that agents travel longer trajectories until settling in the aggregation (more dense paths). However, for bigger values of opening angle (C and D), a consensus can be achieved faster since the agents traveled less (more sparse paths). Nevertheless, we can see that more segregated groups are formed for bigger values, as pointed out by the blue circles in Figure 3.12-D. This segregation can be explained by the fact that increasing the opening angle also increases the influence of the closest neighbors. With that, even a neighbor even located in the "back" of the agent would interfere in its behavior. With smaller opening angles, the agent tends to "ignore" the neighbors more.

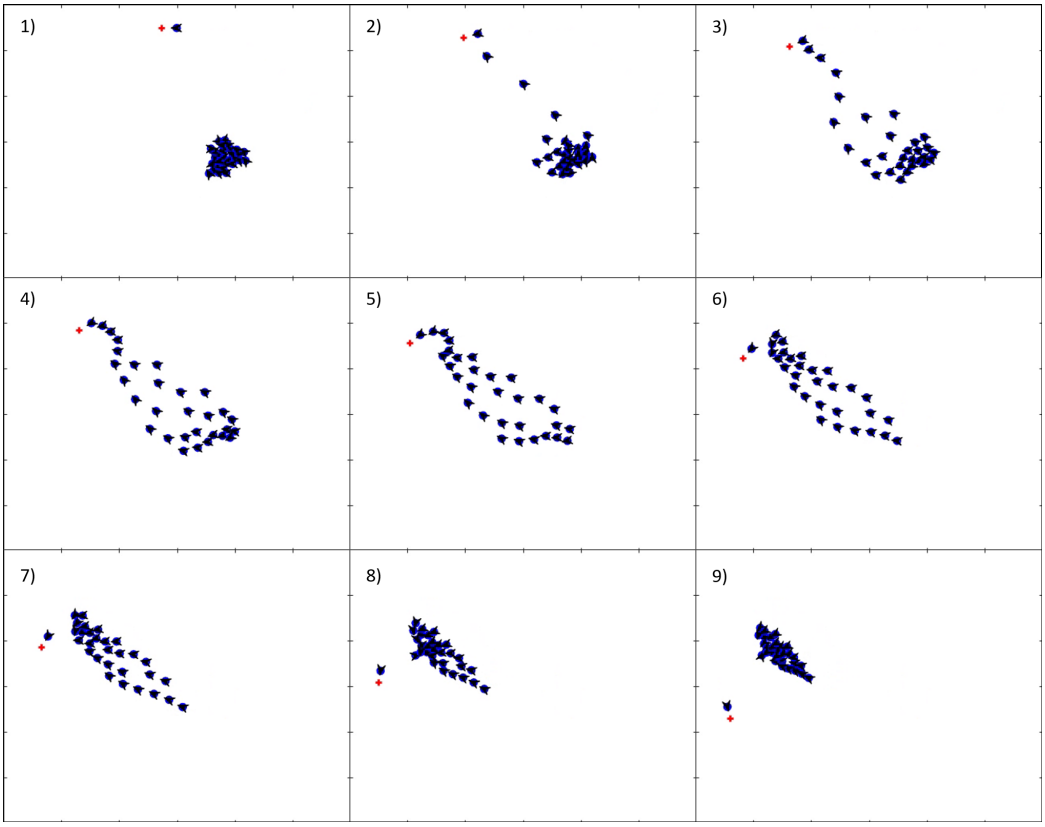


Figure 3.13 – Screenshots for a simulation with 15 flocking agents and one informed.

Informed agents In this scenario, we illustrate the influence of a single agent over the whole. For that, 15 agents are also placed randomly initially; nevertheless, one of the agents is informed about the target position and follow it. The informed agent is the same along with all this episode.

In the first screenshot from Figure 3.13, the flocking agents are all aggregated, while the informed agent is following the target (red cross). In the second screenshot, we can see that some of the agents started to be influenced by the informed one, and gradually all the agents start to migrate. In the last screenshots, they all stop in a new aggregation. This pattern was observed several times in this simulation.

Note that, although the group movement is not uniform, one agent alone is enough to influence the positioning of the whole flock. We will see in the next simulation that the presence of more informed agents are sufficient to obtain a more uniform flocking movement.

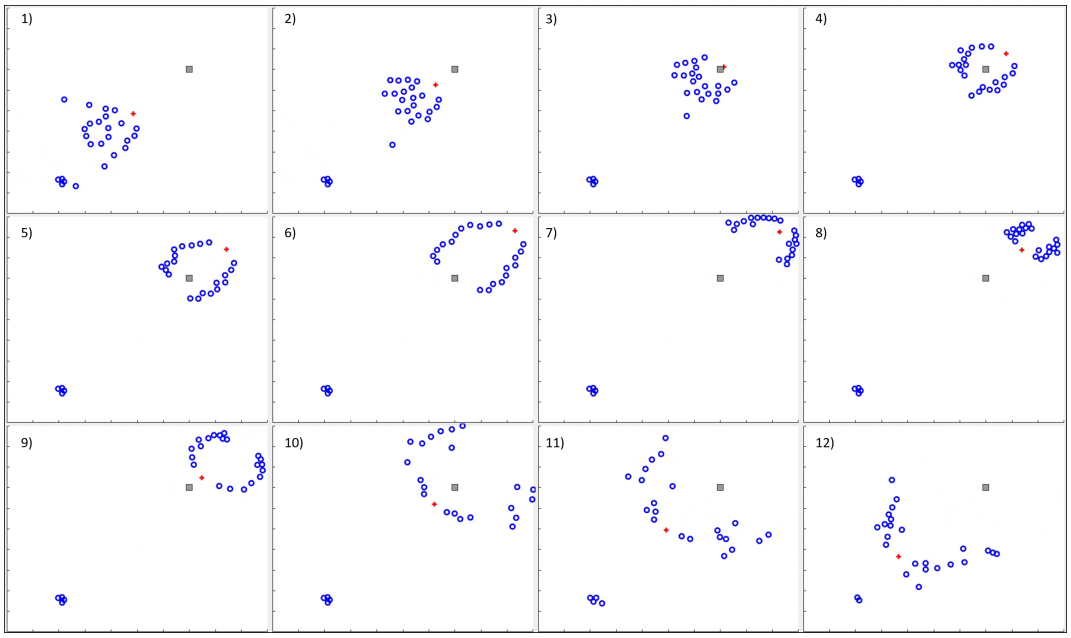


Figure 3.14 – Screenshots for generalized scenario: target tracking (red cross), obstacle avoidance (grey square) and flocking.

Cruising trajectory and obstacle avoidance In this last scenario, we propose to validate the generalized guidance law (3.19), where all agents have the same tendency of flocking, avoidance and pursuit. For that, 25 agents have been initialized in the bottom left part of the arena, with the following configuration of field-of-vision: $\beta = 1.2\pi$ and $R_F = 25$. The target is traveling in a linear trajectory with a constant velocity. Figure 3.14 shows the target (red cross) starting at the center of the arena and moving towards the up-right corner.

Henceforth, there are no chosen informed agents, but all agents tend to pursue the target if it is inside their field-of-view. Furthermore, target and obstacle (grey square) are also perceived inside only when inside the FoV.

Figure 3.14 we can see in screenshots 1 and 2 that the flock is cohesively following the target. Once the target overtakes the obstacles, screenshot 3, the agents progressively split into two groups. The two groups keep segregated until the obstacle does not interfere anymore in the formation, and gradually the two groups start to join again (screenshot 11 and 12).

3.4 Conclusion

In this chapter, we went beyond the classical pursuit, and we investigated three complementary scenarios. We have tried different assumptions on the target's observation and different missions as well. Following a summary of the propositions of this chapter.

In the section **Pursuit Target Prediction**: we suppose that all agents knew the target's position and velocity during the pursuit. With that, we proposed implementing a predictive scheme to foresee the future location of the target. Furthermore, all the pursuit was then based on the prediction, and the positioning of the agents was based on the error estimated prediction error. We saw a consistent improvement in the performance compared to the simple Group Deviated Pursuit (Chapter 2). Although the need for target's global information is hard to implement in real-life, this methodology can be used as "ground-truth" to evaluate the previous pursuit approach, from Chapter 2. Furthermore, qualitative analysis in the pursuit's approaches will be provided in Chapter 5.

In the section, **Pursuit with non-cooperative agents**: a reactive algorithm for pursuit and collision avoidance for non-cooperative agents was proposed. We considered the scenario where a single pursuer had to track a target in an environment shared with non-cooperative agents. The navigation algorithm was obtained using the geometrical rule of parallel navigation and the pursuer target tracking and obstacle avoidance properties. The algorithm was validated in simulations for three different scenarios, including when the pursuer navigates into an environment where non-cooperative agents are evolving.

Finally, in the section, **Flocking**: we investigated the cohesive collective movement, and we conciliate it with the pursuit and avoidance previously seem. This new flocking approach is based on parallel navigation and dispenses the use of "alignment" or velocity consensus terms, which is one advantage for real-time implementation since the neighbors' velocity is not easy to be measured by ordinary onboard sensors. We validate our proposition through simulations, where the scenarios of aggregation, independent traveler, and cruising with obstacles were analyzed.

Chapter 4

Implementation and Experiments

This chapter describes the experiments and the implementation of the pursuit strategies, from the previous chapters, into real time robots. Firstly, we present the quadcopter model, which will be used along with the chapter. Then, we propose a motion control hierarchy for a drone pursuer. This multi-layer controller is responsible for transforming the high-level command from the guidance law (GLs) into the low-level quadcopter inputs. Following, we discuss the hardware platform, exposing the used robots, the software architecture, and the installations. Next, we describe our work on state estimation, in which we implemented a Kalman filter scheme for improving position and velocity estimation. Finally, we expose proof-of-concept results for group-pursuit with quadcopters, implementing several proposed strategies in real time.

Contents

4.1 Drone model

For the experimental validation, we choose the conventional quadcopter configuration, as illustrated in Figure 4.1. Due to its simplistic mechanics and capability to hover and perform indoor flying, the quadcopter has been a frequent prototyping choice in UAV research. Although the high-level guidance law's output could be applied into a vast range of vehicles, we have specially dedicated to quadcopter once it has been the source of criminal usage, as reported in Chapter 1 Furthermore, this configuration is also a recurrent choice for an anti-drone drone (ADD), as we pointed out in section 1.1.2.

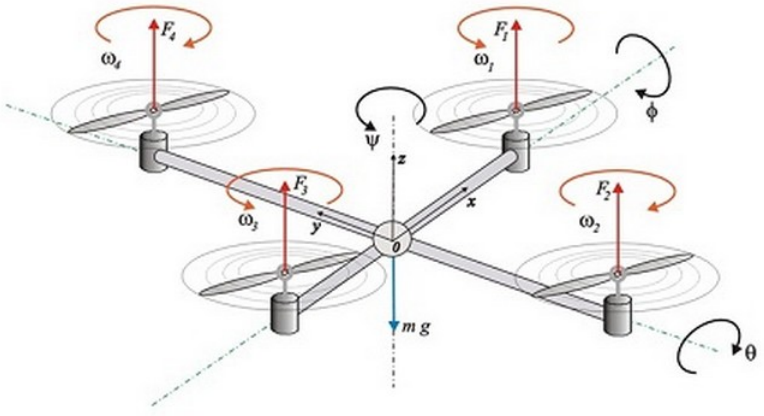


Figure 4.1 – Body frame of the quadcopter.

As exposed in [171], the quadcopter dynamical model can be obtained by representing it as a solid body evolving in a three-dimensional space under the action of the main thrust and three torques, see Figure 4.1. Therefore, two simplified drone models will be used along with this chapter, one in the inertial frame and the other in the body frame. The first one will be used in the task positioning estimation, where we are interested in the drone location in an inertial frame. The second one will be used in the drone's motion control since the inputs from the GLs are given in the body frame.

4.1.1 Inertial Frame

Let us consider the following dynamical model:

$$\begin{aligned}
\ddot{x} &= -\sin(\theta)\frac{1}{m}u \\
\ddot{y} &= \cos(\theta)\sin(\phi)\frac{1}{m}u \\
\ddot{z} &= \cos(\theta)\cos(\phi)\frac{1}{m}u - g \\
\ddot{\phi} &= \dot{\theta}\dot{\psi}\left(\frac{I_y - I_x}{I_x}\right) - \frac{I_r}{I_x}\dot{\theta}\Omega + \frac{l}{I_x}\tau_\phi \\
\ddot{\theta} &= \dot{\phi}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{I_r}{I_y}\dot{\phi}\Omega + \frac{l}{I_y}\tau_\theta \\
\ddot{\psi} &= \dot{\theta}\dot{\phi}\left(\frac{I_x - I_y}{I_z}\right) + \frac{l}{I_z}\tau_\psi
\end{aligned}$$

where the position coordinates are x, y, z and the orientation is given by the Euler angles; roll (ϕ), pitch (θ) and yaw (ψ). The distance between each motor to the gravity center of the vehicle is denoted by ℓ. The inertia of the vehicle in each axis is defined by I_x, I_y and I_z while the inertia of the motor is represented by I_r, the speed of the rotor is defined by Ω and u is the main thrust, τ_i defines the torque control input.

Chain of integrators Therefore, the main dynamics in its linear form to be studied are the altitude, the longitudinal, and the lateral movements. Without loss of generality, it is well known that the altitude dynamics can be represented by two integrators in cascade and the longitudinal/lateral dynamics by four. This is a very common simplification, and it is justified since considering smalls values for the orientation; where the trigonometric function can be simplified as follow: $sin(x) \approx 1$ and $cos(x) \approx x$. Thus, the longitudinal drone's dynamic can be expressed as

$$\ddot{x} = -\theta, \tag{4.1a}$$

$$\ddot{\theta} = u_2 \tag{4.1b}$$

In this cascade representation the control input u_2 is equivalent to τ_θ. Similarly, the lateral dynamics of a quadcopter can be described as a chain of four integrators:

$$\ddot{y} = \phi, \tag{4.2a}$$

$$\ddot{\phi} = u_3 \tag{4.2b}$$

In this cascade representation the control input u_3 is equivalent to τ_ϕ.

Finally, the vertical dynamic can be described by a chain of two integrators, $\ddot{z} = u$, where u is equivalent to the main thrust.

4.1.2 Body frame

Considering the conventional quadcopter configuration showed in Figure 4.1. We use the Newton-Euler approach, where, the mathematical equations for the quadcopter in the body frame can be written as:

$$\begin{aligned} m\dot{\boldsymbol{v}} &= \boldsymbol{F} + \mathbb{R}^T \boldsymbol{F_g} \\ \dot{\boldsymbol{\eta}} &= \mathbb{B}(\boldsymbol{\eta})\boldsymbol{\Omega} \\ \mathbb{J}\dot{\boldsymbol{\Omega}} &= \boldsymbol{\tau} - [\boldsymbol{\Omega}]^\times \mathbb{J}\boldsymbol{\Omega} \end{aligned} \tag{4.3}$$

The bold letters represent vectors. $\boldsymbol{F}$ denotes the thrusts generated by the motors, $\boldsymbol{F_g}$ is the vector force of gravity. $\boldsymbol{\eta} = [\phi_i \ \theta_i \ \psi_i]$ is the vector of Euler angles, $\boldsymbol{\Omega} = [\omega_x \ \omega_y \ \omega_z]$ means the angular velocity in the body frame. m indicates the mass of the drone and $\boldsymbol{v} = [v \ u \ w]^T$ defines the velocity vector of the aerial robot in the body system, $\mathbb{R}$ describes the rotation matrix generated in the order yaw-pitch-roll. $\mathbb{B}(\boldsymbol{\eta})$ represents the matrix that relates the angular velocity and the derivative of the Euler angles. $\mathbb{J}$ is the inertia matrix of the drone. $[\boldsymbol{\Omega}]^\times$ means the skew symmetric matrix of angular velocity and $\boldsymbol{\tau}$ defines the torques applied to the vehicle.

4.2 Motion control hierarchy

4.2.1 Introduction

In Chapters 2 and 3 we proposed several multi-agent strategies in a planar environment. in this initial setup, for simplicity, we considered very simple the kinematics model, also known as the uni-cycle model. Simplified setups are very common in multi-agents, navigation guidance, and differential game analysis. Nevertheless, they do not provide a straightforward implementation into real-systems once they do not consider the robots' dynamic and robots' dimensions.

In this section, we will discuss the extension of the agent behavior into an aerial vehicle, considering the application of a quadcopter. In a few words, the robot behavior is given by a multi-layer controller, as illustrated in Figure 4.2.

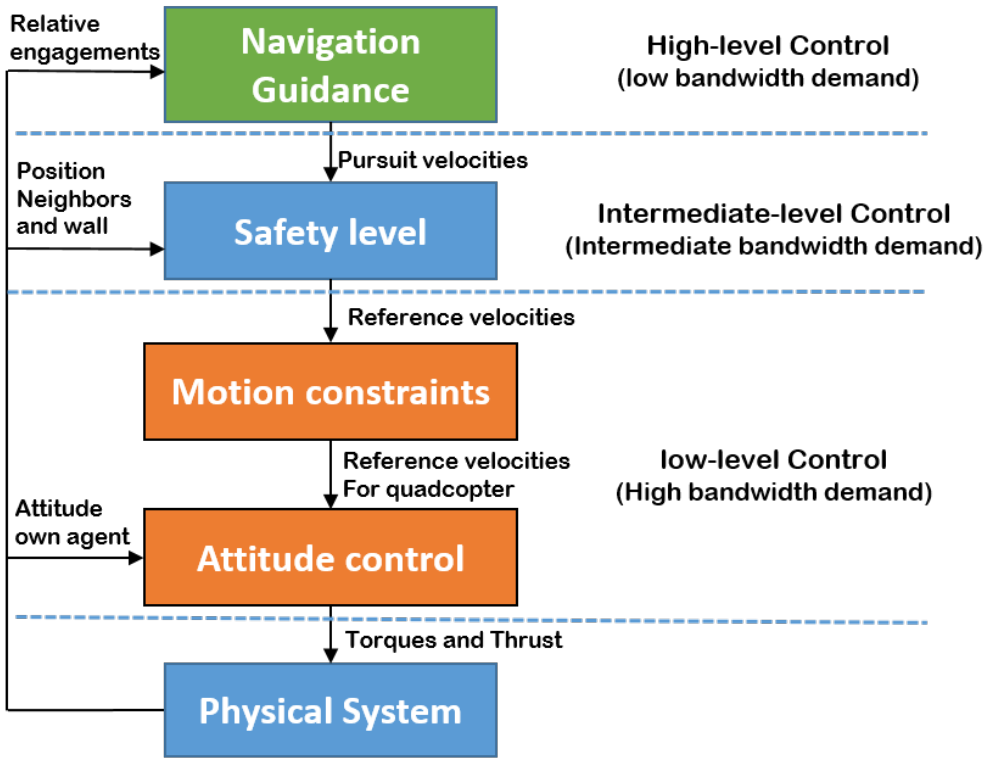


Figure 4.2 – Motion control hierarchy of a quadcopter using guidance laws.

This multi-layer control has a similar composition with the motion control hierarchy used by marine surface vessels, as the one described in [75]. In the higher level is located the Guidance Law, which gives the velocities and turn rates inputs f_{vi}, f_{ui}, f_{wi} and $f_{\dot{\psi}i}$. Following, we have the *Safety constraints* level, which are responsively to add virtual constraints such as delimited workspace, maximal velocities and close-range collision avoidance. The command from this level are expressed by f_{vi}^{saf}, f_{ui}^{saf}, f_{wi}^{saf}, and $f_{\dot{\psi}i}^{saf}$. Next, the *Motion constraints* level is a crucial component which will provide drone's compatible commands, $f_{\theta i}$, $f_{\phi i}$, $f_{\psi i}$ and $f_{\dot{z}i}$, which in this case are the expressed in angular position. Finally, the *Attitude control* is the lower level control, which gives the thrust and torques inputs to the physical system u_1, u_2, u_3 and u_4. in the following, more detailed description about each layer is given.

4.2.2 Higher layer - GL

The pursuit algorithms (or the guidance laws) are responsible for the high-level, or navigational control. It is basically composed by four command laws: f_{vi}, f_{ui}, f_{wi} and $f_{\dot{\psi}i}$, which are, respectively, the references for forward, and lateral velocities, vertical acceleration and angular rate in yaw. Nevertheless, the planar guidance proposed in Chapters 2 and 3, proposes only guidance law for the turn rate $f_{\dot{\psi}i}$, since they consider linear velocity constant $f_{vi} = constant$, and non-holonomic constraints in the 2D plan, resulting in $f_{ui} = 0$ and $f_{\dot{w}i} = 0$.

One exception is for the pursuit strategy based on Reinforcement Learning, where besides the turn rate, law for forwarding velocity f_{vi} is also provided. Furthermore, as will see further in the section 4.2.4, an extension for the vertical plane pursuit will be proposed, and a law for the vertical displacement $f_{\dot{w}i}$ will given.

4.2.3 Safety layer

This layer is responsible for the material's integrity during the experimental validation and under laboratory conditions. Therefore, additional assumptions are considered, such as knowledge of all agents' position and velocity in the

inertial frame. With this complete knowledge of the system, we can assure collision avoidance by adding several virtual constraints, such as delimited workspace, maximal velocities, and close-range collision repulsion. Although the ambushing trajectory, generated by the pursuit algorithms (higher layer), has collision avoidance term, they can not guarantee full collision-free once they do not consider all degrees of freedom, dynamic, and dimensions of the robot application.

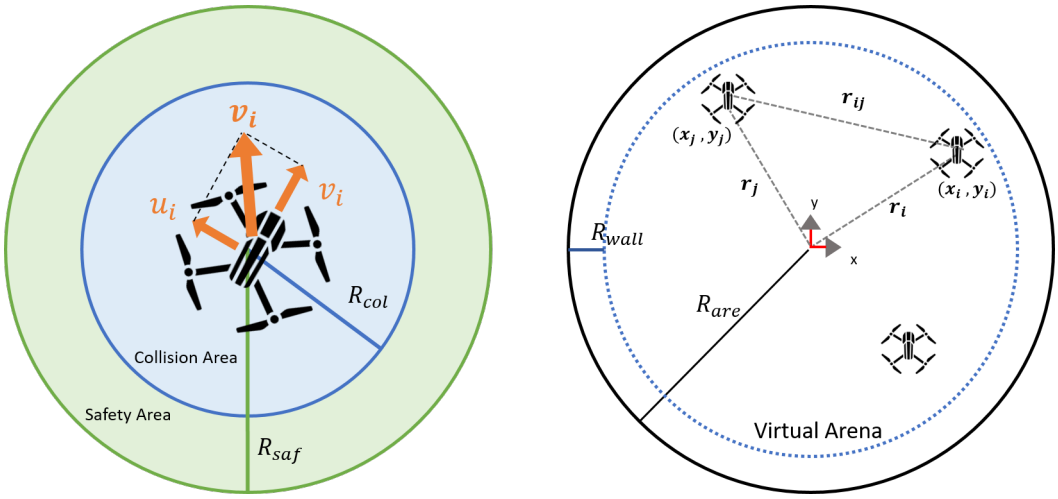


Figure 4.3 – Scheme of the safety level. On the left side, a representation of a drone, with its velocities and its safety radii R_{saf} and R_col. On the right side, three drones and their position vectors are represented inside a virtual arena of radius R_{are} with the thickness of R_wall.

Before going into details, let us consider the illustration from Figure 4.3. In the right picture is illustrated three drones inside a circular arena of radius R_{are}, and the origin of the inertial frame is the center of the arena. Also, $\boldsymbol{r_i}$ and $\boldsymbol{r_j}$ are the position vector of the the agents i and j. Furthermore, in the left picture, a drone with its body velocities (v_i and u_i) and two concentric areas are illustrated. The inner circle is the "collision imminence area," characterized by the radius R_{col}. If a neighbor is inside this area, the repulsion will be maximized. The outer circle is the "safety area," characterized by the radius R_{saf}. The safety layer will ignore all neighbors outside this zone once they propose no collision risk.

The *Safety layer* is composed by the following laws: f_{vi}^{saf}, f_{ui}^{saf}, f_{wi}^{saf}, and

$f_{\psi i}^{saf}$. For computing the law for the velocities, $\boldsymbol{f}_{vi}^{saf} = [f_{vi}^{saf} \; f_{ui}^{saf}]^T$ we borrow the concepts of [53] and [54], where their "flocking law" was given by the sum of several virtual forces: repulsion, friction, and virtual barriers. Nevertheless, in our case, the virtual forces are active only when objects are inside the safety area, such as indicated in Figure 4.3.

Without further ado, the safety law for the velocities is computed as following:

$$\boldsymbol{f}_{vi}^{saf} = \boldsymbol{f}_{i}^{pur} + \boldsymbol{f}_{i}^{wall} + \sum_{j \neq i} \left(\boldsymbol{f}_{ij}^{rep} + \boldsymbol{f}_{ij}^{fric} \right) \theta \left(R_{saf} - |\boldsymbol{r}_{ij}| \right) \tag{4.4}$$

where $\boldsymbol{f}_{vi}^{pur} = [f_{vi} \; f_{ui}]$ is the velocity reference from the higher level (guidance law). It is important to notice that as in the guidance law consider non-holonomic constraints, there is no lateral velocity, consequently $f_{ui} = 0$. Besides, $\boldsymbol{f}_{ij}^{rep}$ and $\boldsymbol{f}_{ij}^{fric}$ denote the repulsion and friction components for each neighbors. The therm $\boldsymbol{f}_{vi}^{wall}$ is the component of repulsion from the virtual limits, the "virtual walls", as illustrated in Figure 4.3-Right. Also, $\boldsymbol{r_{ij}}$ is the vector connecting agents i and j. And finally $\theta(x)$ is a function that the output is 1 if $x \geq 0$ and 0 if $x < 0$.

The repulsion component from an agent j towards another agent i is computed as follow

$$\boldsymbol{f}_{ij}^{rep} = \frac{D(\boldsymbol{r}_{ij} - R_{saf})}{|\boldsymbol{r}_{ij}|} \theta(R_{saf} - |\boldsymbol{r}_{ij}|), \tag{4.5}$$

where D is a coefficient for strength of the repulsion.

For its turn, the friction component from two agents is calculated as follow

$$\boldsymbol{f}_{ij}^{fric} = C \frac{(\boldsymbol{v}_j - \boldsymbol{v}_i)}{(max\{R_{col}, |\boldsymbol{r}_{ij}|\})^2}, \tag{4.6}$$

where C is the coefficient of alignment, which tends to align the agents' velocity as explicated in Chapter 1. In [53], the authors pointed out the importance of the alignment term to "damp" the agents' response for the repulsion forces in interaction. In contrast with the upper layer (GL), this layer requires knowledge of neighbor velocity in the inertial frame, which is justifiable considering experimental validation laboratory conditions.

Finally, the "virtual wall" component are given by:

$$\boldsymbol{f}_i^{wall} = sig\left(|\boldsymbol{r}_{ij}|, R_{are}, R_{wall}\right)\left(\frac{\boldsymbol{r}_i}{|\boldsymbol{r}_i|}\right), \tag{4.7}$$

where R_{are} is the radius of the arena, and R_{wall} is the thickness of the virtual barrier, as illustrated in Figure 4.3-Right. Besides, $sig(.)$ is a Sigmoid function, which can be defined as

$$sig\,(x, R, d) = \begin{cases} 0, & \text{if } x \in [0, R] \\ \frac{\sin\frac{\pi}{2}(x-R)-\frac{\pi}{2}+1}{2} & \text{if } x \in [R, R+d] \\ 1, & \text{if } x > R+d \end{cases} \tag{4.8}$$

Please notice that the safety velocity, such as calculated in (4.4) is only a requirement for experimental validation in laboratory conditions, where workspace is limited, and the drones travel very close to each other. In a real-world application, this safety velocity based on flocking will be much less needed since the pursuer's runnable area is much greater than its "safety area."

4.2.4 Motion constraints

The third level is responsible for transforming the safety velocity input from the upper levels into commands compatible with the robotic platform, which in this case are commands for the angular position: $f_{\theta i}$, $f_{\phi i}$, $f_{\psi i}$ and acceleration in $z-axis$ $f_{\dot{z}i}$.

However, the robot may have more degree of freedom than the GL's output, which is clearly the case of the quadcopter equations (4.3), and the the kinematics model (3.16) considered in Chapters 2 and 3. Therefore, to adapt the quadcopter to the GL commands, two motion constraints model are proposed: *non-holonomic* and *heading-to-target* constraints.

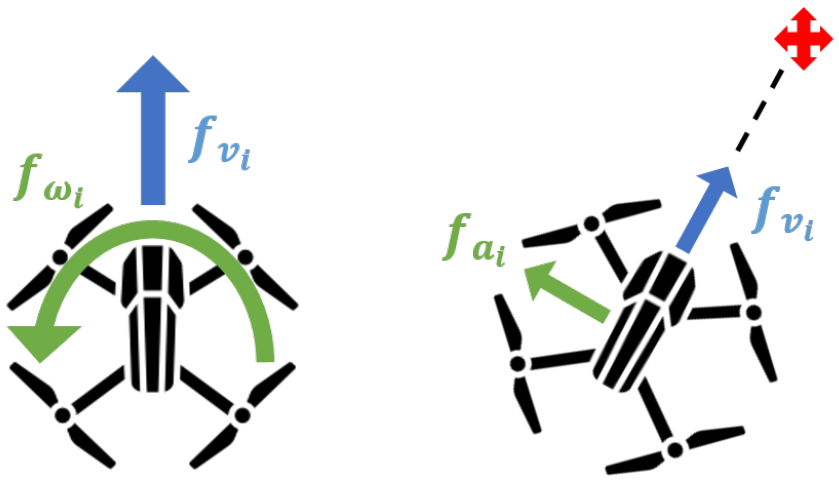


Figure 4.4 – Representation of the two proposed quadcopter's motion constraints: a) non-holonomic b) heading-to-target

Non-holonomic constraints

Possibly the most intuitive way to apply the GL commands into quadcopter configuration would be to impose the non-holonomic constraints into it. It can be done by nullifying the drone's lateral velocity and forcing it to act only in turn around the z-axis (ψ) and in the forward velocity. With that, the quadcopter states, exposed in (4.3) will have the following tendency:

$$v_i \to f_{v_i}^{saf}, \quad u_i \to f_{u_i}^{saf}, w_i \to 0, \quad \omega_{zi} \to f_{\psi_i} \tag{4.9}$$

Considering the quadcopter's simplified model (4.2), and applying the same simplification into the the model 4.3, we can see that $\dot{v}_i \approx -\theta_i$ and $\dot{u}_i \approx -\phi_i$. Then, a simple law to assure the condition 4.9 can be:

$$\begin{aligned} f_{\theta i} &= K_\theta (f_{vi}^{saf} - v_i), \\ f_{\phi i} &= K_\phi (f_{ui}^{saf} - u_i), \\ f_{\psi i} &= \int_0^t f_{\dot{\psi}_i} dt, \end{aligned} \tag{4.10}$$

Note that, for instance, we are not yet considering the vertical displacement.

A section dedicated to the vertical pursuit plan will be given further, see section 4.2.4. For the moment, we consider that the quadcopter is already stabilized in altitude and its vertical velocity is null ($v_z = 0$).

This motion constraint has an interesting property while considering the frontal camera as a navigation sensor: the drone will displace only in the direction of the field of vision, the sensed area, tightly coupling the perception to the movement. For the same reason, this kind of drone motion is commonly observed in FPV piloting, where the only information for the pilot is the frontal camera image, Figure 4.5. In this case, the remote-pilot tends to cancel the drone's lateral drift by combining a roll movement while turning the drones heading.

Figure 4.5 – Illustration of an FPV drone race: once the pilots (left picture) have only the front camera information as perception, they tend to drive the quadcopters in a non-holonomic way.

Heading-to-target

In this second motion type, the drone will regulate its heading always toward the target ($\psi_i \to \lambda_{iT}$) and will keep a positive velocity (v_i) in that direction. The NGL command will be given as lateral acceleration f_a, perpendicular to the LOS. In the quadcopter, the lateral acceleration in the body frame is represented by the state $\dot{u}$.

$$v_1 \to f_{v_i}^{saf}, \quad \dot{u}_i \to f_a, w_i \to 0, \quad \psi_i \to \lambda_{iT} \tag{4.11}$$

This technique's advantage is to improve the target's perception once it is always in the center of the image plane. Furthermore, it allows a straightforward implementation for GL that provides the command in terms of normal acceleration of f_a, rather than the turn rate of f_ψ.

Considering the application of the PNG in terms of f_a (3.11), once the lateral acceleration of the drone is already perpendicular with the line-of-sight (LOS), this configuration allowed a direct implementation of the law, which is also called True Proportional Navigation (TPN) [36].

Altitude controller

In order to establish a guidance controller for the vertical plane, let us at first consider the three-dimensional engagement between a pursuer and an evader as proposed in Figure 4.6. The straight line connecting them, the Line-of-sight (LOS), is denoted by r_{iT}. The angle formed by the LOS and the inertial axes x and y are denoted here by λ^v_{iT} and λ^h_{iT}, respectively. The vectors v_i and u_i are the straightforward velocity and the upward velocity in the body frame. ψ_i represents the yaw of the drone i, and $\hat{\theta}_i$ is a virtual pitch determined by the composition of the v_i and w_i.

Therefore, let us consider a Deviated Pure Pursuit (DPP) guidance law for the vertical plan:

$$f_{\hat{\theta}_i} = K_p * \left(\hat{\theta}_i - \lambda^v_{iT} - \beta\right), \tag{4.12}$$

where $\hat{\theta}_i$ defines a pitch angle and K_p a positive gain. The LOS angle now is defined in the vertical plane (λ_{iT}) and the offset angle is called β.

Since for a quadcopter, the displacement in the vertical plan is function of the thrust $\boldsymbol{F}$ (see Eq. 4.3) it is necessary a function to relate $\boldsymbol{F}$ to the turning rate output $f_{\hat{\theta}_i}$ from (4.12). For that, we consider a virtual pitch that is equivalent to the angle between the vector $\boldsymbol{v}$ and the horizontal plane. It can be given by

$$\hat{\theta}_i = \tan^{-1}\left(\frac{v_i}{w_i}\right) \tag{4.13}$$

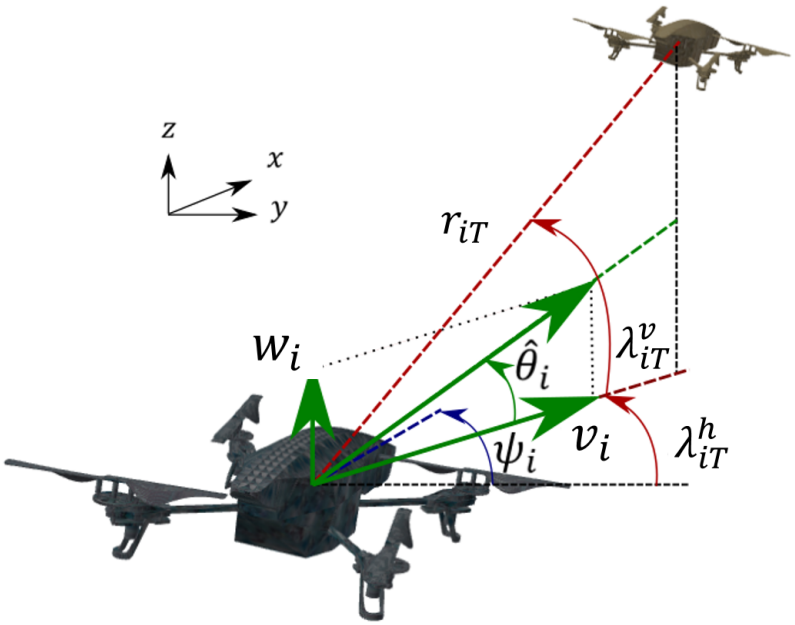


Figure 4.6 – Three dimensional geometry between two quadcopters.

For safety navigation, we consider that the straightforward velocity of the pursuer v_i is changing so slowly that can be considered constant by some time instants; therefore, the derivative of the expression the above becomes

$$\dot{\hat{\theta}}_i = \frac{1}{1 + (\frac{w_i}{v_i})^2} * \left(\frac{\dot{w}_i}{v_i}\right) \tag{4.14}$$

where $\dot{w}_i$ is the vertical acceleration of the pursuer i. Therefore, from (4.14), and considering $f_{\hat{\theta}_i}$ as the virtual pitch rate, it is clear to write

$$f_{\dot{w}i} = \left(v_i + \frac{w_i^2}{v_i}\right) * f_{\hat{\theta}_i}, \tag{4.15}$$

where, $f_{\dot{w}i}$ is the control input for the vertical acceleration.

For avoiding singularity, v_i is imposed to have a minimum value in the interception part. Note from the above that the thrust control is not based on regulation with the target altitude, but with its relative polar coordinates, which can be easier obtained from an on-board camera.

4.2.5 Attitude Controller

The lower-level control is responsible to generate control inputs for thrust (u_1) and torques (u_2, u_3 and u_4) for the quadcopter, following the reference given by the upper level control. This level requires high-frequency processing and is the one most dependent on the vehicle's dynamics.

Therefore, we choose a non-linear controller based on saturation function, proposed in [170], to stabilize the lateral dynamic of the quadcopter 4.2 and follow the angular references. The control law for longitudinal dynamic is given by:

$$u_{2_i} = -\sigma_1(k_1(\theta_i - f_{\theta i})) - \sigma_2(k_2\dot{\theta}_i), \tag{4.16}$$

where, u_{2_i} is the control input for τ_θ, and k_1 and k_2 are positive gains. Besides, $\sigma(.)$ is a saturation function, previously defined in (2.7).

In a similar manner a control law for lateral dynamic can be given by:

$$u_{3_i} = -\sigma_3(k_3(\phi_i - f_{\phi i})) - \sigma_4(k_4\dot{\phi}_i), \tag{4.17}$$

where k_3 and k_4 are positive gains.

The control input for the thrust can be directly obtained from from (4.15), $u_{1_i} = f_{\dot{w}i}$. However, for a pursuit in a fixed plan, we can given an altitude reference (z_i^{ref}) and apply a similar non-linear controller such as:

$$u_{1_i} = -\sigma_a(k_a(z_i - z_i^{ref})) - \sigma_b(k_b\dot{z}_i), \tag{4.18}$$

where k_a and k_b are positive gains.

Finally, control law for the turn around the z-axis is given by:

$$u_{4_i} = -\sigma_5(k_5(\psi_i - f_{\psi i})) - \sigma_6(k_b\dot{\psi}_i) \tag{4.19}$$

where k_5 and k_b are positive gains.

The proof of convergence of the above control laws and more details about its implementation can be checked in [170].

4.3 Material

In this section, we described the hardware used for the experimental validation. We describe firstly the indoor flight arena, the localization system, and the used drones. Finally, we briefly explain the adopted framework, and we expose the software architecture.

4.3.1 Hardware

Flight arena For real time validation, we had at disposal two flight arenas, one indoor and another outdoor. However, except for a few experiments with data-fusion, most of the results exposed here were done in the indoor flight arena. Figure 4.7 shows a picture of four drones inside the indoor arena, also called "volière."

Figure 4.7 – Picture of a group-pursuit experiment in the indoor flight arena.

The volière area is about 100 m^2 with $6m$ in height. Besides, it is equipped with 24 infrared cameras, which composes the motion capture system (Opti-Track), allowing a precisely drone's localization during the flight tests.

AR Drone 2 It is a commercial quadcopter developed by Parrot, Figure 4.8-left. It was chosen due to several advantages, such as low-price, robustness, and easy maintenance. The prototype has already several sensors embedded from the factory. It supports an average of 400g of payload, which can be used by a GPS receiver for some applications. In this work, the drones had the original firmware erased, and they have been "flashed" with FL-Air compatible firmware, developed at Heudiasyc. This operation allowed low-level operations once we could have access, for example, to raw data from the sensors and write PWM signal directly to the motors.

Figure 4.8 – Parrot AR Drone 2 (left) and Parrot Bebop 2 (right). The two commercial quadcopters used for the experimental validations.

Bebop 2 It is another commercial quadcopter also manufactured by Parrot, Figure 4.8-right. It weighs 500 g and offers 25 minutes of flight time. It has an interesting configuration: an onboard computer (a dual-core processor with CPU quad-core), ultrasound and pressure sensors, 3-axis gyroscope, accelerometers, and magnetometer GNSS chip-set. Similar to the previous one, it is also compatible with FL-Air. However, along with this thesis, we have used the Bebop under two configurations, using FL-Air and using its original set-up. For this last one, original firmware and communicating with a remote computer using the *bebop autonomy* SDK [110], which support ROS (Robotic Operating System) [112].

4.3.2 FL-Air

The Framework libre Air, available in [113], is a framework written in C++ for the development of UAV applications. This tool is composed of different applications running in a Linux system and communicating via socket.

Figure 4.9 – Screenshot of the FL-Air simulator in an outdoor environment.

One important application is the *World Simulator*, which allow a reliable evaluation before using the robotic platform. Furthermore, it has a customizable ground-station, which allows on-line parameters tuning and real time flight data checking. More details about the platform can be found in the web-page [113] or in [114].

Although this frame-work is sufficient for most applications, some high-level projects (involving Neuronal Networks, for example) were not supported by FL-Air. Therefore, since we had different design needs, we chose to use alternatives hardware architectures.

4.3.3 Architectures

The following two software architectures were used along with this thesis. The first one, *onboard computation*, has all the control architecture levels implemented inside the drone. The second one, *remote computation*, has its high-level guidance is calculated by a remote computer.

On-board computation

In this configuration, all layers of the motion control, described in Figure 4.2, are implemented in the embedded processor of the drone. This configuration is a fast solution for simple algorithms, which does not require too much computational resource. Furthermore, it is the standard configuration by using the framework

Fl-Air, and all the code was implemented in C++.

Most of the validation using guidance laws based in geometrical rules (such as in sections 2.2 and 2.3 were implemented in this configuration. Nevertheless, the embedded processor was not sufficient to implement the neuronal network policies for section 2.4, which required the next architecture development.

Remote computation

In this configuration, the high-level motion control is calculated in a remote computer and transmitted to the drone by wireless, using UDP protocol. A ROS node is responsible for managing the Optitrack server and the NN policies. A scheme of this implementation can seem in Figure 4.10.

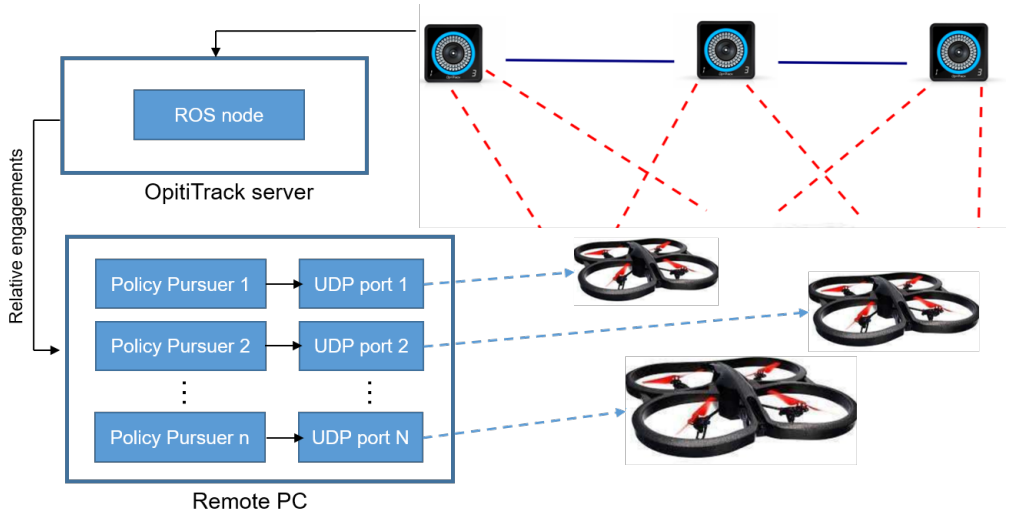


Figure 4.10 – Scheme for the remote computation architecture.

The guidance law based in the Reinforcement Learning (see section 2.4) was implemented by using this configuration. It was chosen considering the embedded processor's low capacity to implement the Neuronal Network (NN) policies generated by the Deep RL algorithm.

4.4 State Estimation

To deal with the state measurement's recurrent problems, such as loss and degradation of data, we implemented a Kalman Filter (KF) scheme to state-estimation. The objective is to have a reliable position estimation considering degraded data from a GPS or *motion capture* system. In this section, we describe the filter design, discussing the models and the composition of the matrices. We propose a dynamic observation matrix C, which would vary according to the data's availability.

These results were published in our work entitled: **Enhanced UAV pose estimation using a KF: experimental validation**. C. de Souza Jr, P. Castillo, B. Vidolov, R. Lozano. Presented in the International Conference on Unmanned Aircraft Systems, Dallas - US, 2018.

4.4.1 Related work and problem statement

One of the main problems for outdoor navigation is to locate with reasonable precision the aerial vehicle. In diverse conditions, such as a vehicle is hovering or close to the buildings, the GPS may not offer the desired precision [120]. Furthermore, the GPS signal can be degraded or lost (non-available) due to a series of reasons as boarded in [121]. Currently, drone applications demand a high precision level of sensory because it is an unstable natural system that demands a high frequency of closed-loop control.

A very popular approach for improving vehicle pose estimation is the Kalman Filter algorithm (KF). Its intensive uses in robotics navigation go from land vehicles, marine robots to spacecraft. The use of KF for attitude estimation in aerospace applications has been extensively explored since the Apollo project (1960) to nowadays. In [119] an interesting review that boarded the evolution and maturation of this application in the aerospace scenario is presented.

For UAV's application in an urban environment, the Kalman Filter has been implemented mainly to improve localization estimation. In [118], the authors implemented a KF together with a complementary filter to lead with the discontinuity in the GPS output while navigating in a big city.

They got interesting results in the pose estimation; however, they only applied it in an open-loop experiment. In [116] and [117], the authors implemented an outdoor navigation controller using an Out-of-sequence KF to deal with the problem of GPS latency. They validated their experiments with outdoors closed-loop flights, although the control architecture is computed in a ground station and not into the drone (on-board).

In this work, a Kalman filter is used for improving UAV pose estimation. Two real cases were studied: loss and degradation of data. The solution is based on fusing data from an INS and a localization system with a linear and low computational cost KF applied to the quadcopter kinematics equations. Several flight tests were realized using a camera motion system emulating some commons problem of a GPS.

The localization problem is pragmatically boarded in this work. We consider low power signals arriving at the drone, consecutively being attenuated and shadowed by buildings or vegetation. For more details about this problem, see [6], where the authors described a detailed overview of propagation problems in GPS.

Hereafter, the degradation is the nomination for the diminution of precision in the data. It is still available meanwhile, in this case, having normal noise. The loss is the non-availability of the data. In our study, it is considered momentarily loss, typically an issue of communication. Both problems are firstly added in simulation using Matlab, and then they are implemented in real time. In this last case, a data problem routine was implemented into the drone firmware, and the ground-station provides its activation with the joystick.

4.4.2 Evolution model

As exposed in section 4.1, the dynamic model of a quadcopter (4.1.1) can be simplified by decoupled chains of integrators. Let us consider the simplified drone's longitudinal dynamic (4.2), which can be represented by chain four integrators. Therefore, (4.2) can be expressed in a state-space representation in a discrete domain:

$$\ddot{\boldsymbol{X}}_{k+1} = A\dot{\boldsymbol{X}}_k + Bu_{2_k} \tag{4.20}$$

where $\boldsymbol{X} = [x, \dot{x}, \theta, \dot{\theta}]$. The discretizated state-space model was done using Ho-Kalman methods as described in [115]. The transition matrix A can be checked bellows:

$$A = \begin{bmatrix} 1 & T_s & -T_s^2/2 & -T_s^3/6 \\ 0 & 1 & -T_s & -T_s^2/2 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Similarly the control matrix can be seen below:

$$B = \begin{bmatrix} -T_s^4/24 & -T_s^3/6 & -T_s^2/2 & T_s \end{bmatrix}^T.$$

In both matrices, the terms concerned with derivatives with order superior to 1 (as $T_s^2/2$ and $T_s^3/6$) are neglected in real time implementation. Since the sample time is short ($T_s = 0.05$), those terms become insignificant in the final results. We consider $u_{2_k} = 0$; nevertheless, the input for the translational movement considered here will be the angular acceleration, which is not measured.

4.4.3 Kalman filter equations

The KF filter is a recursive predictor/estimator of optimal gain primarily used in many engineering sectors. The algorithm is divided into two steps, estimation and prediction. The first stage computes the estimation of the states, $X_{k|k}$, and the covariance error, $P_{k|k}$. The mains equations of this step are

$$X_{k|k} = X_{k|k-1} + K(Y_k - C * X_{k|k-1}), \tag{4.21a}$$

$$\boldsymbol{P}_{k|k} = (I - K * C)P_{k|k-1}(I - K * C)^T + K * R * K^T \tag{4.21b}$$

Using (4.21) the estimation vector $X_{k|k}$ can be estimated taking account the last prediction $X_{k|k-1}$ and the current measurement Y_k. It takes the weighted average based on the Kalman gain K.

For the error covariance estimation $P_{k|k}$ it was used the Joseph's form (4.21)b. For calculating it, the equation takes into account the last value $P_{k|k-1}$, the matrix R (sensor noise covariance), the matrix C (observation), and the Kalman gain.

The second step is characterized by the calculation of the predictions ($X_{k|k+1}$ and $P_{k|k+1}$) and the new updated Kalman gain. Basically, it takes into account the updated estimation ($X_{k|k}$ and $P_{k|k}$) and predicts the next sample step, as the equations below :

$$X_{k+1|k} = A * X_{k|k1} + B * U_k, \tag{4.22a}$$

$$\boldsymbol{P}_{k+1|k} = A * P_{k|k1} + Q, \tag{4.22b}$$

$$\boldsymbol{K} = P_{k|k-1} * C^T * (C * P_{k|k-1} * C^T + R)^{-1} \tag{4.22c}$$

In (4.22), the next step prediction for the states $X_{k|k+1}$ and error $\boldsymbol{P}_{k+1|k}$ are calculated. Besides, the Kalman gain $\boldsymbol{K}$ is updated.

4.4.4 KF matrices and tuning

The complete states vector applied to this filter are $\boldsymbol{X} = [x, \dot{x}, \theta, \dot{\theta}, y, \dot{y}, \phi, \dot{\phi}, z, \dot{z}]$, where ($x$, y, z) denotes the three dimensional position, ($\dot{x}, \dot{y}, \dot{z}$) defines its velocities and (θ, ϕ, $\dot{\theta}$, $\dot{\phi}$) corresponds to the Newton's angles of pitch, roll and its consecutive rates. The yaw angle is considered previously controlled and aligned to the inertial frame.

Evolution model matrix: A describes the drone dynamics in a discrete state-space representation. Details about its development can be checked in the section 4.1. For our dynamics, this matrix has the following form

The **states measurements matrix** Y is the input of the KF, composed by the state's measurements. For knowledge, in the following experimental validation, the localization system's positions and velocities are given by the OptiTrack; besides, the angles and angular rates are provided by the Inertial Measurement Unit (IMU).

$$A = \begin{bmatrix} 1 & T_s & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -T_s & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T_s & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T_s & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -T_s & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & T_s & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The **observation matrix** C, describes the observability of the system. For instance, all the states are considered observable, which results in an identity matrix $C_{10\times10}$. Dynamic changes in this matrix are going to be exposed in the case of data loss.

The **measurements covariance matrix** R is a diagonal matrix (since the sensor errors must be not correlated) composed by measurement variance of each sensor. For example, when using an OptiTrack for positioning, the variance was set as 0.1 m, the orientation angles (using an IMU) to 0.1 rad, and the altitude (using an ultrasound) to 1 m due to its noisy behavior.

The **noisy covariance matrix** Q is harder to determine, and normally its definition is subject to trial and error. Therefore, the covariance for the positions (x, y, z) and the angles (θ, ϕ) were set to a small values (such as 0.1), because in the model, they are results of a simple derivative. The covariance for the velocity ($\dot{x}$ and $\dot{y}$) were set higher values like 10, dues their linearization that gives less reliability in the model. The angular rates were set to a high value as the system's input is unknown by the predictor.

The **initial values** of X (states estimation) were set all to 0 and the initial values of P (error covariance matrix) were set to 100, making P an identity matrix (10x10) multiplied by 100.

4.4.5 Altitude estimation

The following section, which is not present in the original work, presents an alternative for the altitude estimation when barometer and ultrasound sensors are available. It comes from the need to have precise estimation when flying over irregular surfaces and conciliate the measure of two very distinguished sensors.

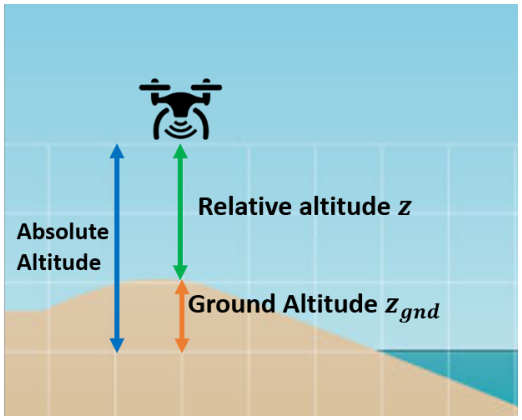


Figure 4.11 – Schema for the altitude estimation, evidencing the states of relative altitude (z) and the ground altitude z_g.

Measurements: The barometer gives a pressure measurement that can be easily converted to an absolute altitude measure ($\tilde{z}_{bar}$). The output of altimeter sensors ($\tilde{z}_{son}$), commonly based in sonar, provides a measurement of the relative altitude z_k. Furthermore, if available, we can also use vertical acceleration measurements from the common accelerometers as the system's input u_{z_k}.

Model: Now, let us consider the following evolution model for the vertical dynamics of a drone:

$$\begin{aligned} z_{k+1} &= z_k + \dot{z}_k \Delta t + \frac{\Delta t^2}{2} u_{z_k}, \\ z_{k+1}^{gnd} &= z_k^{gnd}, \\ \dot{z}_{k+1} &= \dot{z}_k + u_{z_k} \Delta t \end{aligned} \tag{4.23}$$

where z_k^{gnd} denotes the ground altitude and z_k denotes the relative altitude. From this model, the matrices A and B can be obtained, similarly to section 4.4.4.

For its turns, the sensor model can be expressed as:

$$\begin{aligned} \tilde{y}_{son} &= z_k, \\ \tilde{y}_{bar} &= z_k + z_k^{gnd}. \end{aligned} \tag{4.24}$$

From that, the matrix C for the Kalman scheme can be easily determined.

Note that, the relative altitude z_k is being measured explicit by $\tilde{z}_{son}$ and implicit by $\tilde{z}_{bar}$. Allowing the system to have redundancy of an information by considering an intermediate state z_k^{gnd}, which is not relevant for the control of system.

4.4.6 Experimental results

The proposed methodology was applied in real time in our prototype. Several flight tests were carried out with different scenarios. All those experiments were realized in an indoor environment assisted by a motion capture system ($OptiTrack$). It is well known that this kind of sensor measures an object's pose; nevertheless, our idea was to emulate a GPS sensor. Thus, the measures coming from the OptiTrack system were degraded with noise and data lost. Also, the data frequency information was modified to approximate the frequency measurement of a GPS.

The algorithms were implemented in our framework, FLAir, and all the code was written in $C++$. In this architecture, all the algorithms are computed on-board, keeping the ground station just in charge of analyzing the system and acting in emergency cases. The closed-loop system's sample time is on average $5.2ms$, and it is transmitted to the ground station by a local WiFi network also in the same frequency. Three scenarios were boarded: data lost, data degradation, and low-frequency information.

In all cases, four way-points were given as desired coordinates, the sequence is given by $(-3,-3)m, (-3,3)m, (3,3)m, (3,-3)m$ forming a square in the horizontal plane, as one can check in Figure 4.12. In this figure, the first test is showed; here, the quadcopter tracks the desired coordinates in ideal conditions (a good measurement of the position).

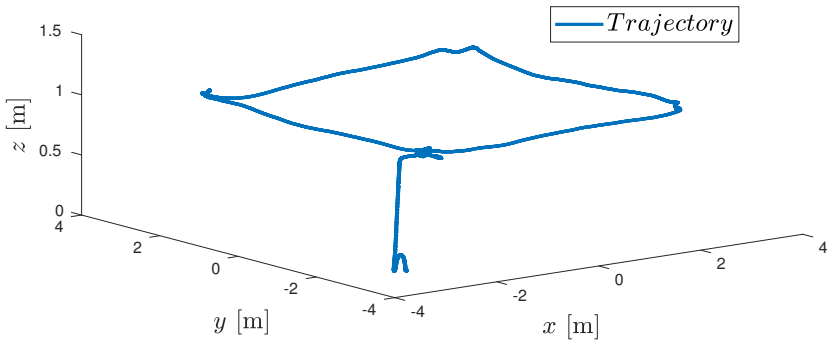


Figure 4.12 – Quadcopter response when tracking the desired points in ideal conditions (no disturbances).

Data lost

In this first scenario, the drone losses for some periods its position information. As explained in the previous section, the loss is detected, and the observation matrix was adapted for a while. For our experiment, the indoor system is "advertised" by a flag ($UAVlost$) in the complete loss of the camera tracking, then it was used as criteria to modify the system.

As results of this scenario, the two following graphs expose the filtering estimation in Figures 4.13 and 4.14. The first one, Figure 4.13, successive loss of 1 second on average were applied to the drone while it was flying. The data loss was applied manually through the ground station. During that, the filter tends to estimate those not observable states more based on the model. We can observe its linear behavior of velocity and position estimated. For brevity, only graphs in the x coordinate are introduced; nevertheless, similar performance is obtained for the other coordinates.

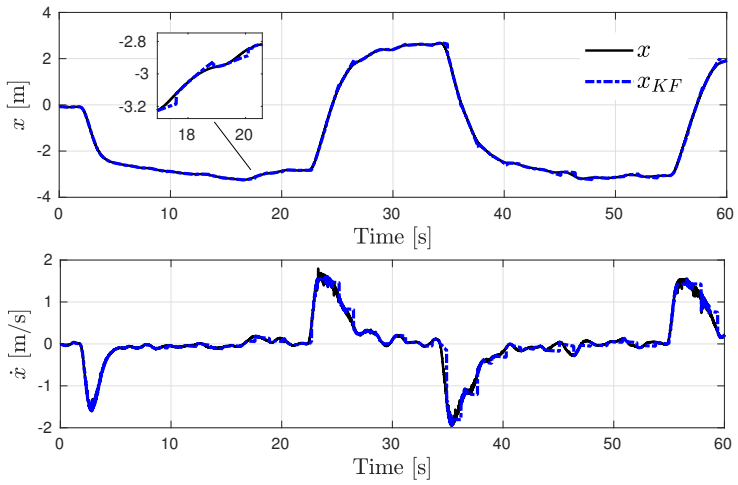


Figure 4.13 – Quadcopter performance with a KF when applying simulated data lost.

The following graphs in Figure 4.14 exposes results of the same case but with a real complete fail from Optitrack. A tunnel was improvised, seeking to deteriorate and cause a complete loss of the cameras' target. Observe in graphs that the positioning system keeps the last possible position till the object recovers it. It is also clear to verify the overshoot caused by the velocity provided by the system. It happens since the positioning system does not measure the velocity but provides it by a discrete derivative of the position. Notice that even if the quadcopter lost its position, the KF can estimate it and ensure a stable behavior in the closed-loop.

Position degradation

This second scenario considers the degradation only in the position data. Although the *Optitrack* system also provides an estimation of velocity, in this case, it was not degraded. Gaussian noise with a mean of 0 and a standard deviation of $5m$ was added to the OptiTrack position measurement, see Figure 4.15. The noise was implemented using a standard class template distribution for $C++$, and the ground station activates it. As explained previously, the noise was applied while the drone was following the desired coordinates.

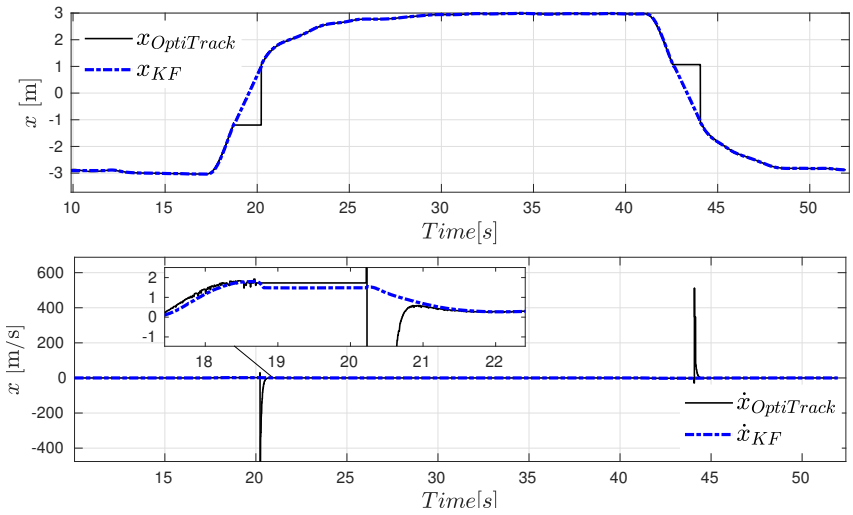


Figure 4.14 – Quadcopter performance with a KF crossing a tunnel. The localization system is totally obstructed for 1.5s in average (equivalent to 300 sample time).

Observe from Figure 4.15 that when adding the noise, the data coming from the positioning system is severally degraded (solid gray line), and when applying the filter, the position is recovered.

The strategy for dealing with data degradation was identifying the signal's quality and adapting the covariance matrix. In real applications, this quality information can be provided by the quantity of camera (in the OptiTrack case) or, in the GPS case, by another quality indicator.

Figure 4.16 illustrates three quadcopter performances: The first one, the vehicle response in ideal conditions and non-KF (solid red line). The second one, the black dotted line indicates the quadcopter response when tracking the desired coordinates with data degradation and without KF. Finally, the blue dashed line presents aerial vehicle behavior with data degradation and using the KF. Notice here that the performance is recovered. Finally, the blue dashed line presents aerial vehicle behavior with data degradation and using the KF. Notice here that the performance is recovered.

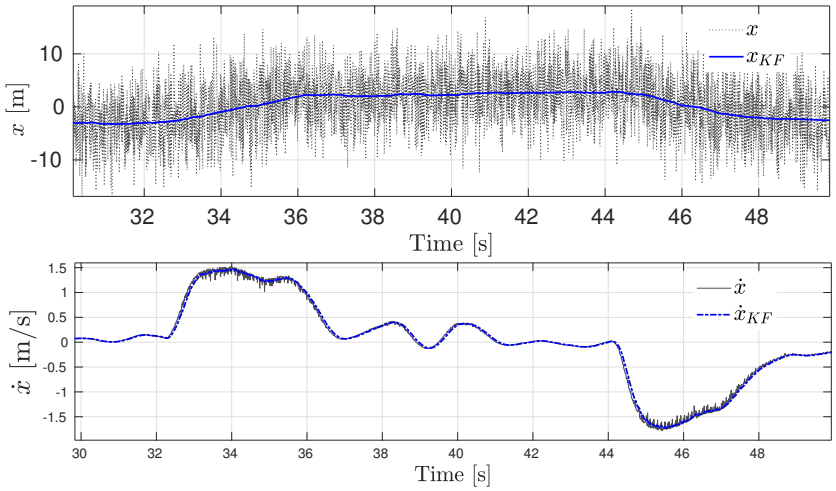


Figure 4.15 – Quadcopter performance with a KF applied to degraded position data.

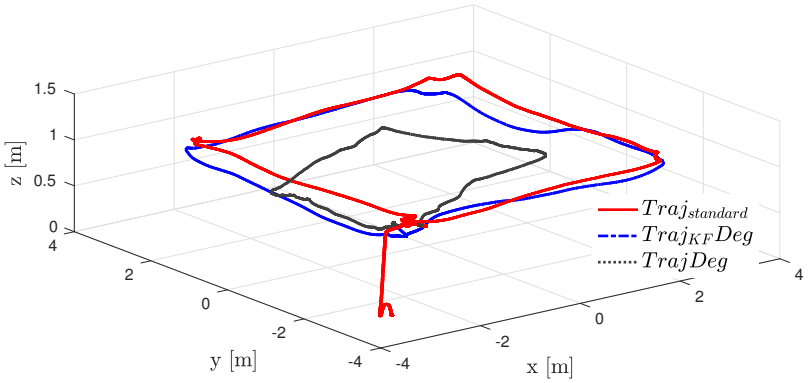


Figure 4.16 – Quadcopter response using KF for position data degradation. The solid red line is the standard performance (no disturbance), black dotted, and blue dashed lines respond in case of degradation. The first one without KF, and the second one is applying the algorithm.

Low frequency information

In this scenario, a non-synchronized measurement Kalman Filter is proposed. The idea is to emulate a GPS performance in real applications since the arrival time, and the frequency of the sensors (GPS and IMU) are not synchronized. Being the *GPS* frequency from 5 to 10Hz and the IMU close to 200Hz.

Our strategy was to set pose as non-observable in matrix C, when there is no GPS available. With that, the system keeps the estimation basing on the IMU input and the evolution model. The position information given by the OptiTrack system is upgraded with a frequency of $3Hz$. This frequency value is normally even lower than a standard one provided by a low-cost GPS for drones.

Note in Figure 4.17 how the KF recovers the position in a case of low frequency and no synchronized GPS data.

Some experiments of this work are available in a video in the following link: *https://youtu.be/k0FobmF7Fv0*

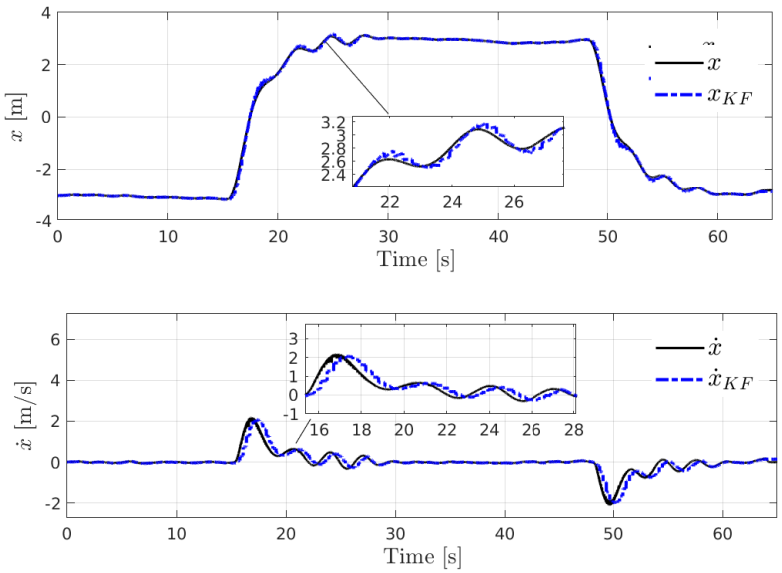


Figure 4.17 – Quadcopter response in a case of low frequency data information. The position provided with 50Hz by the OptiTrack was reduced 3Hz.

Altitude estimation

In this last scenario, we expose the results obtained by implementing the KF scheme exposed in Figure 4.18. The experiments were conducted in the outdoor arena, and the quadcopter had to take over, staying in a position hold for a few seconds and then landing again. The plot with the measurements and the estimation are available in Figure 4.18.

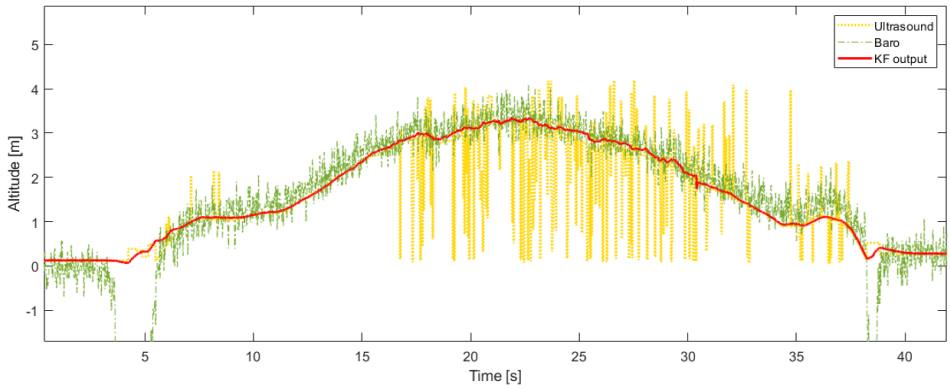


Figure 4.18 – Quadcopter taking over and landing after few seconds of hover. The altimeter measurements $\tilde{z}_{son}$ are very imprecise when the quadcopter is flying above two meters.

In Figure 4.18 we can see that the altimeter provides a very noisy measurement when the drone is flying above 2 meters approximately. In contrast, the barometer, although less precise, does not suffer from this altitude effect. Finally, we can see by the fusion of these two sensors that a smooth estimation can be obtained by applying the KF scheme section 4.2.4.

4.5 Experiments with Pursuit

In this section, we present an experimental validation - proof of concept - for several group-pursuit strategies. The validated strategies namely are *Group Deviated Pursuit* from Chapter 2, *Three dimensional Pursuit* from section 4.2.4 and *Reinforcement Learning Pursuit* from Chapter 2. However, all of those high-level guidance law were applied in the the real time quadcopter by using the multi-layer control scheme, proposed in section4.2.

4.5.1 Group Deviated Pursuit - GDP

Three scenarios are proposed to validate experimentally the DPG (2.5). In the first one, the intruder drone remains in a fixed position, and three pursuers drones track it. In the second scenario, a single evader travels toward the target, passing by two static obstacles. Finally, in the last scenario, the target, moving with a user's random movements, is tracked by three pursuers aerial drones.

Three pursuers and one static target

In this first experiment, a target is placed in the ground with a distance of $6m$ from the pursuers. The pursuers are placed in an initial triangular formation with a distance of approximately $2m$ from their closest neighbors. This experiment aims to reproduce the simulation pattern (see Figure 3.7). The parameters used in this test are presented in Table 4.1.

Parameters	Scenarios 1 and 3	Scenario 2
α	20^o	10^o
u_{max}	1.3	1.3
R_{cap}	1.6	1.6
$Field-vision$	360^o	120^o
R_{int}	5	2
R_{col}	1.5	0
a_0	1	0

Table 4.1 – Control parameters used in the real time scenarios.

From Figure 4.19, the obtained result for this scenario can be observed. It is clear that the hunting pattern described in Figures 2.4 and 4.19 is reached. Nevertheless, the irregular trajectories observed in the pursuer's agents are caused by the airflow produced by the drones' proximity and the absence of control position of each one.

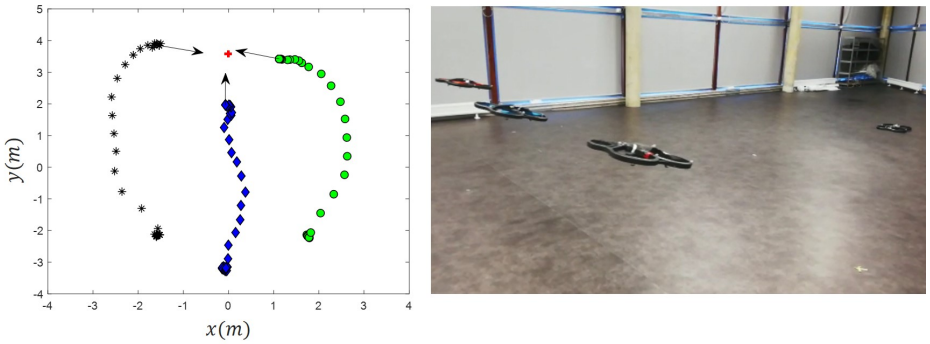


Figure 4.19 – real time performance when three aerial drones pursuit a fixed target. The first image (left) shows the displacement of the drones in the flight arena. The second one is a picture of the experiment.

One pursuer, two obstacles, and one static target

In this scenario, we consider just one drone in the bottom-left side of the flight arena and a fixed target located in the upper-right position, as can be seen in Figure 4.20. Two static obstacles were placed in the trajectory from the drone to the target. The goal of this experiment is to highlight the obstacle avoidance properties included in the proposed algorithm. The parameters used in this experiment are done in Table 4.1. The agents "detect" the obstacles when they are located inside an interaction radius, $R_{int} < 2m$ and inside of its field-of-vision of 120^o. These parameters are different from those used in the cooperative pursuit, where the range-of-interaction is bigger, and no limitation of the field of vision (360^o) is imposed.

The difference in scenario 2 concerning the others is due for two reasons; firstly, the idea is to use onboard cameras to detect the obstacles with a limited vision and range field. Second, impose a small deviation angle, α, is enough to

reproduce a collision-free trajectory in a pure obstacle avoidance case, while in the cooperated pursuit, bigger deviation angles are necessary to increase the covered area by the fleet. In Figure 4.20, the algorithm performance, when tracking a static target with obstacle avoidance, is illustrated. Notice here the good performance of the control scheme.

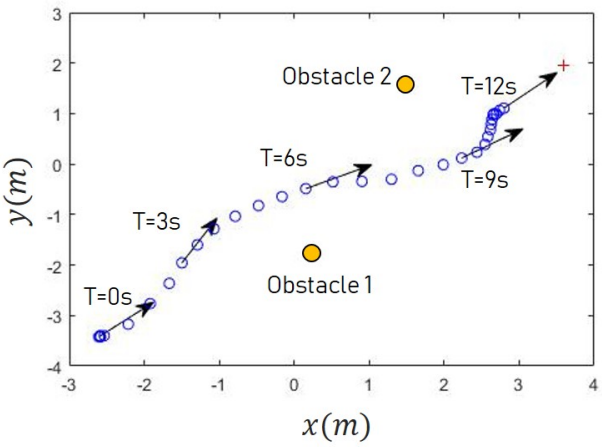


Figure 4.20 – real time behavior when an aerial drone pursuits an aerial intruder drone in presence of static obstacles.

Three pursuers and one faster intruder

In this last scenario, an operator manually controls the target, trying to get away from the three pursuers. In contrast to the scenario presented in simulations, here, the goal for the pursuers is not to intercept the target but to keep a safe distance from it. Besides, to ensure our drones' security and for limitations of the workspace, the target will fly in a different plan than the pursuers, choosing $z_d = 0.8$m for the intruder and $z_{d_i} = 1.7$m to the pursuers. The parameters for this flight are also exposed in Table 4.1.

In Figure 4.21, some snapshots are introduced to illustrate aerial drones' performance pursuing the faster intruder. This figure was done using real data coming from the experiment; the figure is composed of 9 screen-shots taken with an interval of $2.75s$ from one to the other.

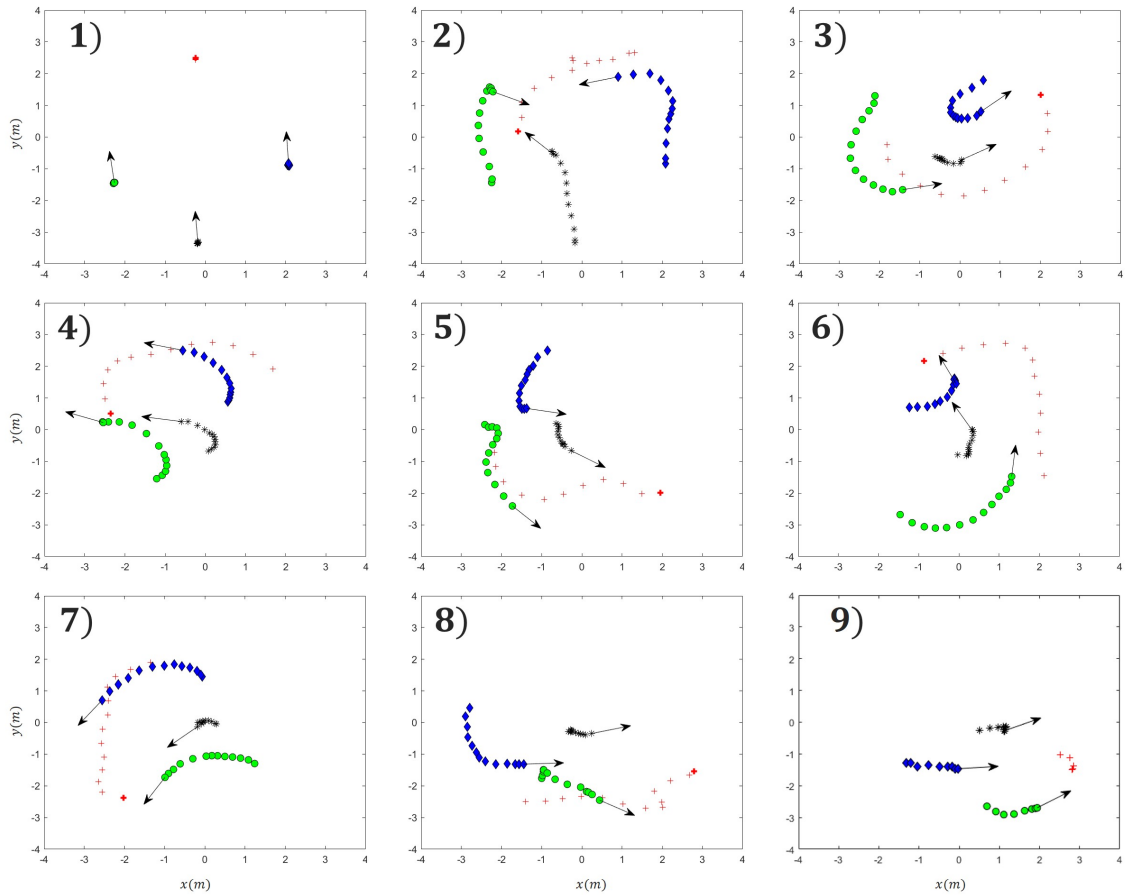


Figure 4.21 – real time performance when three aerial drones pursuit an aerial intruder drone with random movements.

Each agent's position was plotted ten times for each screen-shot, with a sample time of $0.275s$. The evader's superior velocity ($2m/s$) is highlighted by the distance traveled by it in each screen-shot.

The collision avoidance property was implemented only in the pursuers. Since the evader was piloted manually, there were no guarantees of free-collision, and for this reason, the target flew in a different plane from the pursuer. Nevertheless, the virtual wall was the only property implemented into the evader to avoid crashes and plotting purposes. These flight tests can be appreciated in : https://youtu.be/Kr3UFtZ-dd8

We can appreciate the evident cooperative hunting pattern that emerged during these flight tests. Therefore, even facing a complex pursuit, the pursuer's drones could track and intercept the evader's movements with a faster and reactive target. Note that none formation is intended to be kept; instead, the drones re-arrange between themselves to corral the intruder. Observe that the roles of positioning (center, left, and right-wings; as exposed in Chapter 2 are constantly changing in the function of the fleet's relative position to the target.

Moreover, taking into account the size of the virtual arena (imposed to 6×6m) and the presence of 4 drones flying inside, this approach has demonstrated a robust, practical performance in the disturbances produced by the airflow of the aerial vehicles.

4.5.2 3D Pursuit

The 3D pursuit is implemented using two decoupled GLs: the GDP, from section 2.2 for the horizontal navigation, and the Vertical DPP, from section 4.2.4, for the navigation in the vertical plane.

Three experiments were proposed to validate these strategies. In the first two ones, we aimed to validate the vertical guidance law (4.15) alone, and in the last one, we proposed a three-dimensional pursuit with two pursuers and an evader.

Altitude control

This first real time experiment was carried-out to verify the performance of the vertical guidance law (4.15) with only one pursuer and one intruder in two different scenarios, one for fixed and other for variable offset angles. For both cases, the initial position of the pursuer and target is given by: $(x_i, y_i, z_i) = (-2, 0, 0.8)$m and $(x_t, y_t, z_t) = (5, 0, z_{t_d})$m, where z_{t_d} will change for each case.

In the first test, see Figure 4.22, the target (red crosses) was placed with three different altitudes, $z_{t_d 1} = 0.2$, $z_{t_d 2} = 1$ and $z_{t_d 3} = 1.7$ all in meters. The black crosses indicate the pursuer's traveled trajectory, and the green circle with an arrow is its final position and virtual pitch $\hat{\theta}_i$. We can see that in all cases, the pursuer traveled towards the target interception. For security reasons, the

pursuer stops at 1.5 m before the target.

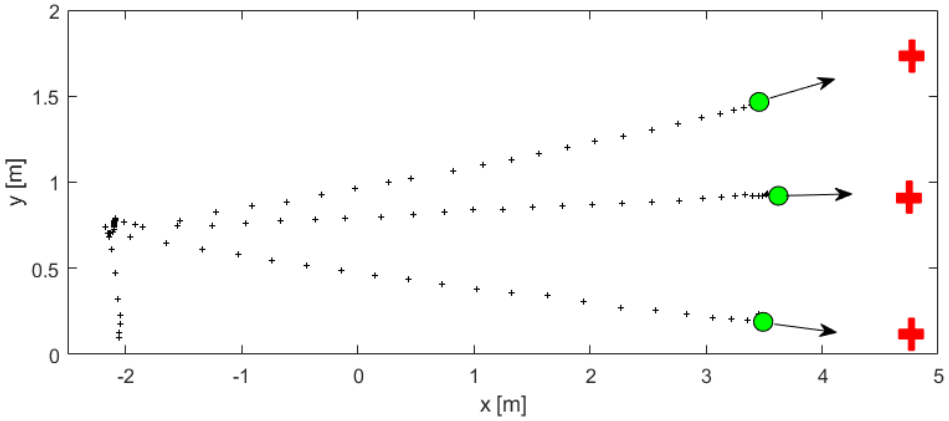


Figure 4.22 – Validation of vertical guidance law (4.15): the black crosses indicate the path traveled by the pursuer (quadcopter) towards the target placed in three different altitudes.

Varying offset

In the second experiment, see Figure 4.23, the target altitude was fixed at $z_{t_d} = 1.8$m, and the control goal is to test the displacement in the vertical plan for different values of α_{iT} angles. Observer that, similar to Figure 2.4, the ambushing behavior also emerged in the vertical plane.

3D pursuit

Once validated the pursuit algorithm's performance for different altitudes, the proposed algorithm is evaluated in real time in a 3D environment. In this scenario, two drones were used as pursuers acting autonomously when applying the altitude control (4.15) and the pursuit algorithm (3.9) with a fixed offset, $\alpha_{iT} = \pi/6$. The target is controlled manually by the user.

In Figure 4.25, the experimental results obtained during the flight tests are presented. The three screenshot samples represent the agents' traveled trajectories in three sequential instants. In this figure, the first line of graphs represents the vertical plane displacement in the x, z-axes, and the second line,

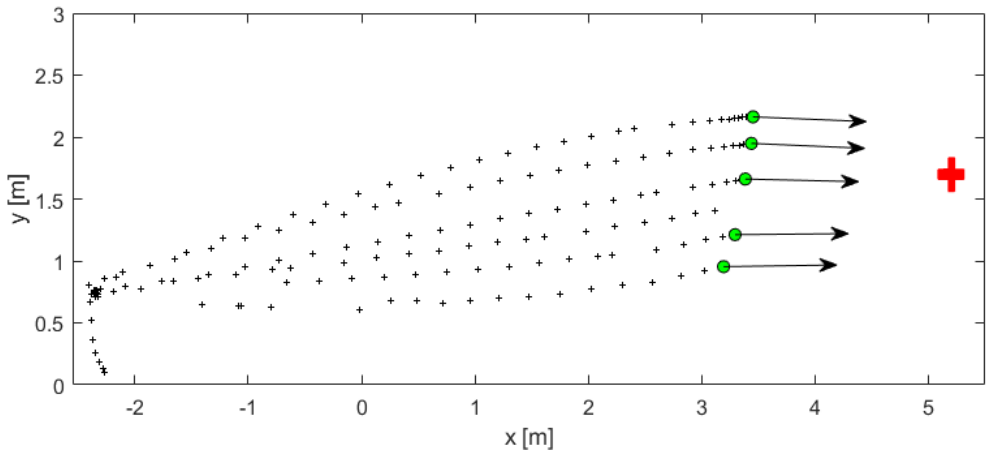


Figure 4.23 – Ambushing behavior in the vertical plan: the black crosses indicates the path followed by the pursuer (quadcopter) towards a fixed target under different values of offset α_{iT}.

the horizontal performance in the x, y-axes. A green circle denotes the pursuer's last position and a black star symbol. Similarly, their last orientation in yaw angle is denoted by the arrows. The red cross denotes the target or intruder. Finally, the reduced size of their symbols represents the trail of the traveled trajectory for all agents.

Notice in these screen-shots that the pattern of ambushing forming in the space is again presented. A video of this experiment is available at:

`https://youtu.be/8CUZZIrBBqA`

4.5.3 Reinforcement Learning

An extract from the real time experiment can be seen in Figure 4.25. In these four sequential pictures, we can see clearly the emergence of the ambushing behavior: the pursuers once close to each other (frame 1), spread toward the target (frames 2 and 3), and finally, they regroup by cornering the target (frame 4).

The results are satisfactory, and they seem even more promising once considering the simplistic conditions of training. Remember that the considered model (3.16) does not consider the non-linear dynamic of the quadcopter during the

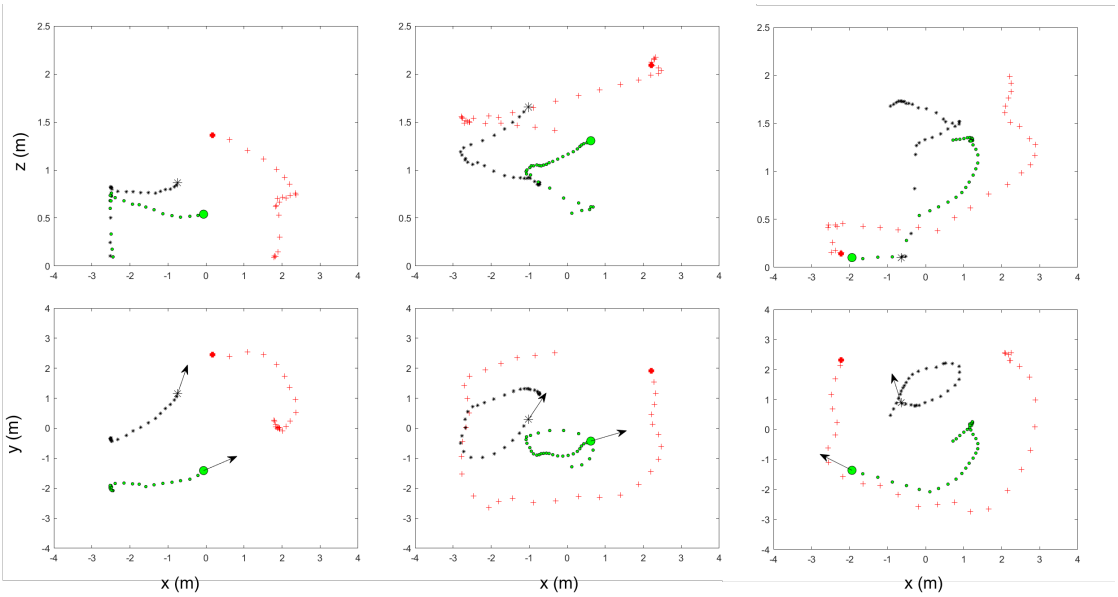


Figure 4.24 – real time three-dimensional pursuit with quadcopters: The path of two pursuers (denoted black star and green circle) and the target(red crosses) are given in these screenshots. The columns denotes three sequential timestamps, while in the each line is exposed the traveled path along $x-y$ and $x-z$ planes

training. However, this evidenced the importance of multi-layer control architecture, where the RL policies can take the high-level decisions of navigation, and a lower-level control can control the vehicle's attitude and assure safety navigation.

This result also arouses the interest in proposing more complex scenarios. The capacity of having a large number of inputs into a Neuronal Network leaves the possibility of including a larger observation vector, which could consider, for example, multi-forms obstacles and no-go zones.

Finally, although pursuit strategies have also been identified, training in a more realistic environment is required for a real application of this technique. For example, the fact that there is a security layer and that the capture does not occur generates the pursuers situations not experienced during the training, which results in inappropriate actions.

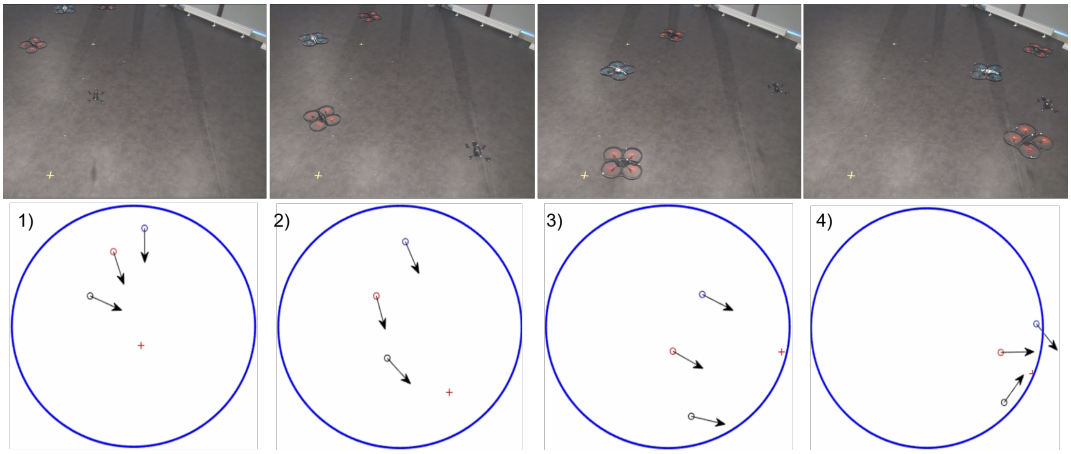


Figure 4.25 – Screenshot for the real time experiment.

4.6 Conclusion

This chapter exposes a complete framework for implementing guidance laws in a group of real time drones. Besides, real time flight trajectory recordings were shown, enabling a qualitative analysis of the pursuit strategies. In the following, we leave to the reader a summary of each section's achievements.

In this first section, we went deeper into the drone model in question, the quadcopter. Consequently, we expose two mathematical representations of a drone's dynamics, one in the inertial frame and the other in the body frame. These models are useful to understand how to apply the guidance laws on this specific robot platform and estimate states.

In section *Motion Control Hierarchy*, we propose a multi-layered control system that aims to take the GL inputs and transforms them into control inputs for the robot platform. To do this, we went first through the safety layer, where collision-avoidance criteria was added to safe the experiment. Next, we propose a layer of "motions constraints" aiming to adapt GL's inputs with the quadcopter model. Finally, we chose an attitude control to generate the thrust and touch commands for the quadcopter.

In section *Hardware*, we expose which materials were used throughout this validation process. Starting from the flight arena and localization system, then exposing the used drones, and finally, giving details of the software architecture.

In section *state-estimation*, we aimed to face a practical problem, where the positioning of drones in outdoor navigation was inaccurate. We studied KF filtering schemes to filter and estimate quadcopter states through inaccuracies and data loss.

Finally, in *Experiments with Pursuit* section, we show results from three implementations of pursuit strategies. In Group Deviated Pursuit, one can see the algorithm's efficiency and the formation of ambush trajectories. Similarly, these results were extended to the three-dimensional space, and similar patterns were also observed in the three-dimensional trajectories. Finally, we have the preliminary results of the implementation of the Reinforcement Learning algorithms.

Conclusions

The intriguing phenomenon of collective motion has been progressively unveiled by the scientific community. Fascinating examples of flocking with autonomous drones and impressive aerial choreographies are already realities, and it has become a common activity in entertainment shows. However, the numerous previous efforts to reproduce the collective motions fail to reproduce the equally fascinating collective hunting phenomenon. Although the last one can be seen as a subgroup of the first, striking differences distinguish between flocking behavior and collective hunt behavior.

As discussed in the introduction, the characteristics of cohesion and alignment, so common in flocking, give rise to dispersal and ambush in collective hunting activity. In view of this theoretical gap, this thesis proposes to model the collective hunting behavior differently, not only seeking to describe the given behavior but also concerning applying such models in real robots.

The interest in artificially reproducing collective hunting is an actual demand from different areas. For example, in biology, ethologists seek to understand and model animal behavior to propose strategies for handling and preserving species. In robotics, the understanding of collective hunting can help perform tracking tasks that would be very complex for a single robot, such as the pursuit and interception of a faster and more agile intruder.

As discussed in Chapter 2, the anti-drone drone has already proved to be a handy tool in combat against civilian drones' unlawful use. However, the pursuit made by a single stalker is already disadvantageous due to a faster or more agile target.

Our approach: The mainline of this thesis tries to reproduce the collective behavior through geometric rules and laws of orientation. We propose multi-agent solutions to the conventional virtual physical laws, common in much of the literature in the area. We show that the ambush behavior can emerge from simple geometric rules such as Deviated Pursuit and Parallel Navigation with offset. Besides, we also investigate multi-agent persecution with deep reinforcement learning techniques.

Furthermore, following this thesis's main methodological line, we investigated problems complementary to the collective pursuit, such as the evasion of dynamic obstacles and different navigation approaches in flocks. Following, a summary of the work carried out in this thesis.

Summary of the chapters

In chapter 1, we carried out a brief but careful review of the main work related to the drone's group chase. First, we present an overview of the current civilian problem with drones, the difficulties of enforcing the law, and the current containment solutions. Among those solutions, we point out the drone anti-drone as a promising one, but holding in its great advantage also its weakness: having the same nature and the threat.

Then, we review the current collective motion literature. We emphasize the importance of bio-inspiration, which was the motivation for the main models of collective movements, and we also reviewed the main multi-agent techniques applied to robots. Then, we introduced the vast pursuit-evasion field, highlighting remarkable works in the area involving the pursuit curves. Besides, we expose the recent group-pursuit field, which tries to reconcile the flocking with classic pursuit analyses.

Finally, we study the classic guidance laws, which are extensive literature deeply related to bellicose navigation but recently extensively applied to robotic navigation. Further, we expose the kinematic models and the geometric engagement used in their analysis, as well as some of the main classic models used in this text.

In Chapter 2, we studied the problem of the group-pursuit of a single-evader in 2D. We propose three different approaches to solve this problem. The first GDP, the simplest of all, is based on the classic deviated pursuit (DPP), where pursuers choose different offsets, depending on their current engagement toward the target and its neighbors. These different angles provide different routes that reproduce ambush trajectories.

The second approach is based on the Parallel Navigation principle, where each of the pursuers aims to maintain a constant bearing angle to the target, which allows for more efficient trajectories. Similar to the previous approach, the offset angle concept was also maintained here, which avoids collision between two pursuers and allows different ambush routes.

Our third methodology aims to build collective hunting behavior based on an automatic mechanism, reinforcement learning, where Neuronal Networks are trained to describe an individual's behavior. In this approach, the pursuers interacting initially at random will discover by trial and error the optimal behavior in this pursuit in a simulated environment.

In Chapter 3, we went beyond the planar group pursuit, and we explored three different scenarios. First, we start with a different pursuit hypothesis, where all agents can know the target's position in the global frame. From this global information, it was possible to use a predictor scheme to estimate the target's future trajectory. Then, we adapted our first deviation chase algorithm (GDP), making each individual chase the target in a future position, and finally, the chase angles are based on estimated prediction errors.

In the second part of this chapter, still a different hypothesis, we study a scenario where a single-pursuer tends to chase a dynamic target with an unknown trajectory in an environment with non-cooperative agents (or mobile obstacles). In this case, we also use the Parallel Navigation principle; the geometric rule, once used in the interception, was now used for collision avoidance.

Finally, we went beyond the pursuit itself, and we presented an approach to reproducing generalized collective motions, such as focusing or flocking, but also through geometric laws.

We propose that from estimating the centroid of the perceived neighborhood and the geometric rule of constant angle, a collective movement's verisimilar behavior can be obtained.

Chapter 4 presents the implementation of the previous algorithms on a robotic platform, the quadcopter. To do this, we first studied the model of the quadcopter dynamic; both defined in the inertial frame and the body frame. Then, we introduced our hierarchical control of movement, composed of four layers.

The highest level is responsible for the orientation law and gives commands at linear and angular velocity. Right after, we have the safety layer, responsible for maintaining the physical integrity of the vehicle. In indoor experimental tests, where collision is not desired, additional safety measures, such as repulsion forces and virtual barriers, have been implemented.

In the sequence, motion control is then implemented to adapt the high-level (and non-holonomic) commands to the drone model (holonomic). Finally, we present some experimental results of the proposed guidance laws implemented in real-time drones. We present screenshots of the pursuers and evaders' trajectories and provide a qualitative analysis of the chase.

Discussion

Pursuit: large vs. small groups This work aims to add a small contribution to the emerging field of group-pursuit, which according to [106], is a fusion between the fields of collective movements (flocking) with classic pursuit techniques. Although the concept of group pursuit has been in place for more than a decade, most work considers predators' behavior to be very minimalist, which differs little from flocking behavior. These models' simplicity aims to apply and analyze the pursuit on a large scale, as in [34] and [153], where teams of hunters of the order hundreds are used. Indeed, these models have great value when analyzing and describing individual organisms' behavior, which in short have a similar order of magnitude.

However, as explained in Chapter 3, this minimalism in the predatory model does not correspond, for example, with the collective hunting behavior of superior mammals, such as the groups of lions or packs of wolves. The differences occur both in terms of the strategy itself and in the number of agents. For example, in several initial works, such as in [153] and [34], the pursuit behavior is purely in aligning its velocity (or orientation) directly to the prey's instant location (Pure Pursuit). In contrast, our proposal focuses on the agent and its interaction with its close neighbors. Furthermore, the pursuit's efficiency is taken into account, along with the formation of an ambush pattern.

Efficiency in pursuit: Similar to our approach, Janosov's work [52] is also concerned with the efficiency of the pursuit. In this case, the model parameters were subjected to optimization processes to improve catch rates. However, the strategy requires knowledge of the prey in global coordinates, which is not ordinarily applicable to the pursuit of non-cooperative targets. Furthermore, this approach is based on virtual forces (attraction and repulsion) and velocities alignment (similar to Vicsek's flocking models [77]) and additional components for the pursuit behavior. Moreover, this large number of parameters also makes tuning work difficult.

As pointed out in Chapter 3, our work has an efficiency very similar to Janosov's work, yet dispensing with hypotheses of knowledge of the target in global coordinates: in the three approaches proposed in this chapter, the target is perceived only in relative and polar coordinates. Besides being realistic and compatible with natural predators, such hypotheses also allow for a more reliable implementation in robotic systems since polar and relative coordinates are consistent with sensors commonly embedded in robots, such as a camera or a Lidar.

Flocking and collision avoidance: Besides the group persecution, this thesis also proposes mild additions to the broader term of collective behavior. The results presented in Chapter 4, although preliminary, point to a possible alternative for reproducing the flocking phenomenon, not based on the sum of artificial physical forces but based on guidance laws and geometric relations.

Once our approach relies only upon local and polar information, our work attempts to contribute to the field of vision-based or communication-less flocking.

One of the main challenges for applying flocking in real-robots is measuring neighbors' orientation or measuring their velocity, which is required in most of the flocking algorithms, such as [49, 58, 77]. Usually, works ignore these two states' measurements, consider robotic systems with very slow dynamics, such as in [139]. In the case of applications with UAVs, as pointed out by [53], the quadcopter's reactivity and dynamics lead very easily to system instability, and knowledge of speeds and orientation is necessary to dampen the system.

In our approach, the alignment is not calculated by the average of the neighbor's velocity, but it naturally occurs when applying the constant bearing and constant range rule, which can be obtained through the range and bearing measurements.

In the field of navigation with uncooperative agents (or avoiding dynamic obstacles), most strategies are also based on the sum of virtual forces, of attraction and repulsion, such as [10], which can work even well under the condition of action of non-dynamic obstacles. However, as they do not consider the dynamics (and kinematics) of the obstacles, and consecutively these techniques do not perform well with mobile obstacles. Another line, such as [15], considers the agents' dynamics/kinematics and predicts the trajectories to determine the imminence or not of collisions. However, these consist of computationally costly approaches. Our strategy aims at a reactive and straightforward way to avoid collisions with dynamic objects, not requiring global knowledge, the position requiring optimization techniques. Again, our hypotheses are based on the relative and polar perception of obstacles, implementing real robots more feasible.

Applicability In addition to this thesis's main problem, the use of drones in defense of civil air space against intruder drones, the models proposed here can be applied in several other tasks. Still, in the field of surveillance, any other task that requires the tracking of a more agile and fast target can be solved with techniques similar to this one.

Another possible area could be the ethological studies in Biology, where models describe animal behavior and better understand a specific animal population's dynamic.

Future work

Perception: For more practical applications, it is necessary to work more deeply with questions of perception—for example, the acquisition and processing of data obtained by camera how to deal. In addition to the challenge of identifying a quadcopter in a moving image, the difficulty is also linked to the distinction between cooperative and non-cooperative agents.

A clue to solve the perception problem would be the fusion of data between position information to be communicated between individuals and the information obtained by the individual's own perception.

Low Level Control: As this is not the main focus of this study, the low-level control part was conducted simplistically. However, to obtain more reactive results, it would be interesting to analyze the low-level reference segment and study more effective and robust control laws for tracking velocities references.

Reinforcement Learning In the RL experiments, the considered agent's model in simulation was pretty straightforward. Certain restrictions on practical applications, such as repulsion due to avoidance of obstacles, were not considered in training and may cause effects not previously known by behavioral policies. A future direction would be to take into account more realistic dynamic models with robotic platforms.

Multi-agent: The works with obstacle avoidance and flocking are still in a very preliminary phase. Deepening in the theoretical field, using metrics and comparisons with previous works are necessary. Besides, more real-time implementations must be made to analyze the capabilities and deficiencies of these approaches.

Bibliography

[1] Sunggoo Jung et al. "Perception , Guidance , and Navigation for Indoor". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 2539–2544.

[2] Christoph Briese, Andreas Seel, and Franz Andert. "Vision-based detection of non-cooperative UAVs using frame differencing and temporal filter". In: *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018* (2018), pp. 606–613.

[3] Robert L Allen. "Quadrotor intercept trajectory planning and simulation". PhD thesis. Naval Postgraduate School, Monterey, CA, USA, 2017.

[4] Divya Agarwal and Pushpendra S Bharti. "A Review on Comparative Analysis of Path Planning and Collision Avoidance Algorithms". In: 12.6 (2018), pp. 608–624. DOI: 10.5281/zenodo.1316879.

[5] Thomas Statheros, Gareth Howells, and Klaus McDonald-Maier. "Autonomous ship collision avoidance navigation concepts, technologies and techniques". In: *Journal of Navigation* 61.1 (2008), pp. 129–142. ISSN: 03734633. DOI: 10.1017/S037346330700447X.

[6] Ruben Nuredini. "Bio-inspired Obstacle Avoidance: From Animals to Intelligent Agents". In: *Journal of Computers* (2018), pp. 146–153.

[7] Juan I. Giribet, Ignacio Mas, and Patricio Moreno. "Vision-based Integrated Navigation System and Optimal Allocation in Formation Flying". In: *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018* (2018), pp. 52–61. DOI: 10.1109/ICUAS.2018.8453429.

[8] Jae H. Lee et al. "Autonomous target following with monocular camera on UAS using Recursive-RANSAC tracker". In: *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018* (2018), pp. 1070–1074. DOI: 10.1109/ICUAS.2018.8453285.

[9] Erwin Perez et al. "Autonomous Collision Avoidance System for a Multicopter using Stereoscopic Vision". In: *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018* 91768 (2018), pp. 579–588. DOI: 10.1109/ICUAS.2018.8453417.

[10] Igor Henrique Beloti Pizetta, Alexandre Santos Brandao, and Mario Sarcinelli-Filho. "Control and Obstacle Avoidance for an UAV Carrying a Load in Forestal Environments". In: *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018* (2018), pp. 62–67. DOI: 10.1109/ICUAS.2018.8453399.

[11] Jose Luis Sanchez-Lopez et al. "Towards trajectory planning from a given path for multirotor aerial robots trajectory tracking". In: *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018* (2018), pp. 1342–1351. DOI: 10.1109/ICUAS.2018.8453428.

[12] Ema Falomir, Serge Chaumette, and Gilles Guerrini. "A 3D mobility model for autonomous swarms of collaborative UAVs". In: *2019 International Conference on Unmanned Aircraft Systems, ICUAS 2019* (2019), pp. 196–204. DOI: 10.1109/ICUAS.2019.8798199.

[13] Lili Ma. "Cooperative target tracking by altering UAVs' linear and angular velocities". In: *2019 International Conference on Unmanned Aircraft Systems, ICUAS 2019* (2019), pp. 69–78. DOI: 10.1109/ICUAS.2019.8797882.

[14] N.A. Shneydor. *Missile Guidance and Pursuit: Kinematics, Dynamics and Control*. Chichester: Horwood Publishing, 1998, p. 286. ISBN: 1898563438.

[15] Matthew J Clark. "Collision Avoidance and Navigation of UAS Using Vision-Based Proportional Navigation". PhD thesis. Embry-Riddle Aeronautical University, Daytona Beach, Florida, 2017.

[16] Stephen A. Murtaugh and Harry E. Criel. "Fundamentals of proportional navigation". In: *IEEE Spectrum* 3.12 (1966), pp. 75–85. ISSN: 00189235.

[17] Fethi Belkhouche, Boumediene Belkhouche, and Parviz Rastgoufard. "Parallel navigation for reaching a moving goal by a mobile robot". In: *Robotica* 25.1 (2007), pp. 63–74.

[18] PA Wilson, CJ Harris, and X Hong. "A line of sight counteraction navigation algorithm for ship encounter collision avoidance". In: *The Journal of Navigation* 56.1 (2003), pp. 111–121.

[19] Samuel T Fabian et al. "Interception by two predatory fly species is explained by a proportional navigation feedback controller". In: *Journal of The Royal Society Interface* 15.147 (2018), pp. 1–13.

[20] Fethi Belkhouche and Boumediene Belkhouche. "A method for robot navigation toward a moving goal with unknown maneuvers". In: *Robotica* 23.6 (2005), pp. 709–720.

[21] Haim Weiss, Ilan Rusnak, and György Hexner. "Adaptive proportional navigation guidance". In: Haifa, Israel, 2018, pp. 365–393.

[22] Caroline H Brighton, Adrian LR Thomas, and Graham K Taylor. "Terminal attack trajectories of peregrine falcons are described by the proportional navigation guidance law of missiles". In: *Proceedings of the National Academy of Sciences* 114.51 (2017), pp. 1–6.

[23] Fethi Belkhouche. "Reactive path planning in a dynamic environment". In: *IEEE Transactions on Robotics* 25.4 (2009), pp. 902–911. ISSN: 15523098.

[24] Yadong Chen et al. "A modified cooperative proportional navigation guidance law". In: *Journal of the Franklin Institute* 356.11 (2019), pp. 5692–5705.

[25] Thierry Lefebvre and Thomas Dubot. "Conceptual design study of an Anti-Drone Drone". In: ***In**: 16th AIAA Aviation Technology, Integration, and Operations Conference* (June 2016), pp. 1–14.

[26] David Solinger, Patrick Ehlert, and Leon Rothkrantz. "Creating a Dogfight Agent". In: *Technical Report DKS05–01/ICE 10* (Apr. 2005).

[27] Daniel Hert. "Autonomous Predictive Interception of a Flying Target by an Unmanned Aerial Vehicle". PhD thesis. Prague, Czech Republic: Czech Technical University in Prague, May 2018. DOI: 10.1115/1.4003863. arXiv: arXiv:1703.06870.

[28] Jasper Thomas Arneberg. "Guidance Laws for Partially-Observable UAV Interception Based on Linear Covariance Analysis". PhD thesis. Massachusetts, USA: Massachusetts Institute of Technology, June 2018.

[29] Hu Yulan, Zhang Qisong, and Xue Pengfei. "Study on multi-robot cooperation stalking using finite state machine". In: *Procedia Engineering* **29** (2012), pp. 3502–3506.

[30] Hiroaki Yamaguchi. "A cooperative hunting behavior by multiple nonholonomic mobile robots". In: *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218*. Vol. 4. IEEE. 1998, pp. 3347–3352.

[31] Zhiyong Wu et al. "A multi-robot cooperative hunting approach based on dynamic prediction of target motion". In: *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE. Macau, China, Dec. 2017, pp. 587–592.

[32] Alfredo Weitzenfeld, Alberto Vallesa, and Horacio Flores. "A biologically-inspired wolf pack multiple robot hunting model". In: *2006 IEEE 3rd Latin American Robotics Symposium*. IEEE. Santiago, Chile, Oct. 2006, pp. 120–127.

[33] Cristian Muro et al. "Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations". In: *Behavioural processes* 88.3 (2011), pp. 192–197.

[34] Luca Angelani. "Collective predation and escape strategies". In: *Physical review letters* 109.11 (2012), pp. 1–5.

[35] Yuan Lin. "Bat swarming as an inspiration for multi-agent systems : predation success , active sensing , and collision avoidance". PhD thesis. Blacksburg, Virginia, USA: Faculty of the Virginia Polytechnic Institute and State University, USA, Feb. 2016.

[36] Neryahu A Shneydor. *Missile guidance and pursuit: kinematics, dynamics and control*. Elsevier, 1998.

[37] Paul Nahin. *Chases and Escapes The Mathematic*. Princeton, New Jersey: Princeton University Press, 2007.

[38] Fethi Belkhouche and Boumediene Belkhouche. "A method for robot navigation toward a moving goal with unknown maneuvers". In: *Robotica* 23.6 (2005), pp. 709–720.

[39] Fethi Belkhouche, Boumediene Belkhouche, and Parviz Rastgoufard. "Parallel navigation for reaching a moving goal by a mobile robot". In: *Robotica* 25.1 (2007), pp. 63–74.

[40] Hamid Teimoori and Andrey V Savkin. "A biologically inspired method for robot navigation in a cluttered environment". In: *Robotica* 28.5 (2010), pp. 637–648.

[41] Ruoyu Tan and Manish Kumar. "Proportional navigation (PN) based tracking of ground targets by quadrotor UAVs". In: *ASME 2013 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers. Palo Alto, CA, USA, Oct. 2013, pp. 157–173.

[42] Haomiao Huang et al. "Guaranteed decentralized pursuit-evasion in the plane with multiple pursuers". In: *2011 50th IEEE Conference on Decision and Control and European Control Conference*. IEEE. Orlando, FL, USA, Dec. 2011, pp. 4835–4840.

[43] Jie Li et al. "Coordinated multi-robot target hunting based on extended cooperative game". In: *2015 IEEE International Conference on Information and Automation*. IEEE. Lijiang, Yunnan, China, Aug. 2015, pp. 216–221.

[44] Anna Zafeiris and Tamás Vicsek. *Why We Live in Hierarchies?: A Quantitative Treatise*. Springer, 2017.

[45] Philip E Stander. "Cooperative hunting in lions: the role of the individual". In: *Behavioral ecology and sociobiology* 29.6 (1992), pp. 445–454.

[46] Jianwei Gong et al. "A GA based combinatorial auction algorithm for multi-robot cooperative hunting". In: *2007 International Conference on Computational Intelligence and Security (CIS 2007*. IEEE. Heilongjiang, China, Dec. 2007, pp. 137–141.

[47] Wei Wang et al. "A rapid hunting algorithm for multi mobile robots system". In: *2007 2nd IEEE Conference on Industrial Electronics and Applications*. IEEE. Harbin, China, May 2007, pp. 1203–1207.

[48] Yong Duan, Xiao Huang, and Xia Yu. "Multi-robot dynamic virtual potential point hunting strategy based on FIS". In: *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC*. IEEE. Nanjing, China, Aug. 2016, pp. 332–335.

[49] Craig W Reynolds. *Flocks, herds and schools: A distributed behavioral model*. Vol. 21. 4. ACM, 1987.

[50] Tamás Vicsek and Anna Zafeiris. "Collective motion". In: *Physics reports* 517.3-4 (2012), pp. 71–140.

[51] Takuya Saito, Tomomichi Nakamura, and Toru Ohira. "Group chase and escape model with chasers' interaction". In: *Physica A: Statistical Mechanics and its Applications* 447 (2016), pp. 172–179.

[52] Milán Janosov et al. "Group chasing tactics: how to catch a faster prey". In: *New Journal of Physics* 19.5 (2017), pp. 1–16.

[53] Csaba Virágh et al. "Flocking algorithm for autonomous flying robots". In: *Bioinspiration & biomimetics* 9.2 (2014), pp. 1–15.

[54] Gábor Vásárhelyi et al. "Outdoor flocking and formation flight with autonomous aerial robots". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. Chicago, Illinois, Sept. 2014, pp. 3866–3873.

[55] C De Souza et al. "Enhanced UAV pose estimation using a KF: experimental validation". In: *2018 International Conference on Unmanned Aircraft Systems (ICUAS*. IEEE. Dallas, Texas, June 2018, pp. 1255–1261.

[56] Daigo Shishika, Justin K Yim, and Derek A Paley. "Robust Lyapunov control design for bioinspired pursuit with autonomous hovercraft". In: *IEEE Transactions on Control Systems Technology* 25.2 (2016), pp. 509–520.

[57] Guoru Ding et al. "An Amateur Drone Surveillance System Based on the Cognitive Internet of Things". In: *IEEE Communications Magazine* 56.1 (2018), pp. 29–35. ISSN: 01636804. DOI: 10.1109/MCOM.2017.1700452. eprint: 1711.10738.

[58] Reza Olfati-Saber. "Flocking for multi-agent dynamic systems: Algorithms and theory". In: *IEEE Transactions on Automatic Control* (2006). ISSN: 00189286. DOI: 10.1109/TAC.2005.864190.

[59] J M Loffi. "Examining Unmanned Aerial System Threats & Defenses: A Conceptual Analysis". In: *International Journal of Aviation, Aeronautics, and Aerospace* 2.4 (2015), pp. 10–11. DOI: 10.15394/ijaaa.2015.1084. URL: https://doi.org/10.15394/ijaaa.2015.1084.

[60] By James Rogers. *The dark side of our drone future*. 2019.

[61] Ana Holligan. *Eagles trained to take down drones*. 2016.

[62] Leon Rothkrantz. "SURVEILLANCE AND PROTECTION BY DRONES". In: *International Conference on Information Technologies (InfoTech-2017)* (2017).

[63] Chen Ke et al. "A survey on guidance law with impact time constraint". In: 2 (2016), pp. 5711–5715.

[64] Timothy H Chung, Geoffrey A Hollinger, and Volkan Isler. "Search and pursuit-evasion in mobile robotics". In: *Autonomous robots* 31.4 (2011), p. 299.

[65] René Vidal et al. "Probabilistic Pursuit – Evasion Games : Theory , Implementation , and Experimental Evaluation". In: *TRANSACTIONS ON ROBOTICS AND AUTOMATION*. Vol. 18. 5. 2002, pp. 662–669.

[66] Selina Pan et al. "Pursuit , Evasion and Defense in the Plane". In: *American Control Conference*. 2012, pp. 4167–4173. ISBN: 9781457710940.

[67] Dave Wilson Oyler. "Contributions to Pursuit-Evasion Game Theory". PhD thesis. University of Michigan, 2016. URL: https://deepblue.lib.umich.edu/bitstream/handle/2027.42/120650/dwoyler%7B%5C_%7D1.pdf?sequence=1%7B%5C&%7DisAllowed=y.

[68] Alyssa Pierson et al. "Cooperative multi-quadrotor pursuit of an evader in an environment with no-fly zones". In: *Proceedings - IEEE International Conference on Robotics and Automation*. 2016. ISBN: 9781467380263. DOI: 10.1109/ICRA.2016.7487151.

[69] Jose Alfredo Guerrero et al. "Mini rotorcraft flight formation control using bounded inputs". In: *Journal of intelligent & robotic systems* 65.1-4 (2012), pp. 175–186.

[70] Jossué Cariño Escobar, Moisés Bonilla Estrada, and Rogelio Lozano. "Cooperative control for load transportation using two PVTOL vehicles with a passivity approach". In: *2017 International Conference on Unmanned Aircraft Systems, ICUAS 2017* (2017), pp. 1385–1391. DOI: 10.1109/ICUAS.2017.7991470.

[71] Zhicheng Hou et al. "Distributed leader-follower formation control for multiple quadrotors with weighted topology To cite this version : Distributed Leader-Follower Formation Control for Multiple Quadrotors with Weighted Topology". In: *10th IEEE System of Systems Engineering Conference*. San Antonio, TX, United States, 2015, pp. 256–261.

[72] Grand View Reasearch. *Commercial UAV Market Analysis by Product (Fixed Wing, Rotary Blade, Nano, Hybrid), by Application (Agriculture, Energy, Government, Media & Entertainment) and Segment Forecasts to 2022*. 2016.

[73] Cristino de Souza Jr, P Castillo, and B Vidolov. "Reactive pursuit and obstacle avoidance based in parallel navigation". In: *ITCS* (2020), pp. 1–7.

[74] Sabine Hauert et al. "Reynolds flocking in reality with fixed-wing robots: communication range vs. maximum turning rate". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2011, pp. 5015–5020.

[75] Morten Breivik and Thor I Fossen. "Guidance laws for planar motion control". In: *2008 47th IEEE Conference on Decision and Control*. IEEE. 2008, pp. 570–577.

[76] Enrica Soria, Fabrizio Schiano, and Dario Floreano. "The influence of limited visual sensing on the reynolds flocking algorithm". In: *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE. 2019, pp. 138–145.

[77] Tamás Vicsek et al. "Novel type of phase transition in a system of self-driven particles". In: *Physical review letters* 75.6 (1995), p. 1226.

[78] Tucker Balch and Ronald C Arkin. "Behavior-based formation control for multirobot teams". In: *IEEE transactions on robotics and automation* 14.6 (1998), pp. 926–939.

[79] RUFUS Isaacs. *Differential Games, SIAM Series in Applied Mathematics*. 1965.

[80] Boldizsár Balázs, Gábor Vásárhelyi, and Tamás Vicsek. "Adaptive leadership overcomes persistence–responsivity trade-off in flocking". In: *Journal of the Royal Society Interface* 17.167 (2020), p. 20190853.

[81] Matthew Turpin, Nathan Michael, and Vijay Kumar. "Trajectory design and control for aggressive formation flight with quadrotors". In: *Autonomous Robots* 33.1-2 (2012), pp. 143–156.

[82] Chih-ming Kung et al. "The fast flight trajectory verification algorithm for Drone Dance System". In: *2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*. IEEE. 2020, pp. 97–101.

[83] Osamah Saif, Isabelle Fantoni, and Arturo Zavala-Río. "Real-time flocking of multiple-quadrotor system of systems". In: *2015 10th System of Systems Engineering Conference (SoSE)*. IEEE. 2015, pp. 286–291.

[84] Jonghoek Kim. "Three-dimensional discrete-time controller to intercept a targeted UAV using a capture net towed by multiple aerial robots". In: *IET Radar, Sonar & Navigation* 13.5 (2018), pp. 682–688.

[85] Atsushi Kamimura and Toru Ohira. "Group Chase and Escape". In: *Group Chase and Escape*. Springer, 2019, pp. 43–75.

[86] Timothy P Lillicrap et al. "Continuous control with deep reinforcement learning". In: *arXiv preprint arXiv:1509.02971* (2015).

[87] Volodymyr Mnih et al. "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602* (2013).

[88] Neryahu A Shneydor. *Missile guidance and pursuit: kinematics, dynamics and control*. Elsevier, 1998.

[89] Manzil Zaheer et al. "Deep sets". In: *Advances in neural information processing systems*. 2017, pp. 3391–3401.

[90] Philip Arthur Johnson. "Numerical solution methods for differential game problems". PhD thesis. Massachusetts Institute of Technology, 2009.

[91] *CODING A DEEP Q NETWORK IN PYTORCH*. https://www.neuralnet.ai/coding-a-deep-q-network-in-pytorch/. Accessed: 2020-04-19.

[92] Maximilian Hüttenrauch, Adrian Šošić, and Gerhard Neumann. *Deep Reinforcement Learning for Swarm Systems*. Tech. rep. 2019, pp. 1–31. URL: http://jmlr.org/papers/v20/18-476.html..

[93] Lin Xu et al. "Multi-agent Deep Reinforcement Learning for Pursuit-Evasion Game Scalability". In: *Lecture Notes in Electrical Engineering*. Vol. 592. Springer Verlag, 2020, pp. 658–669. ISBN: 9789813296817. DOI: 10.1007/978-981-32-9682-4_69.

[94] Ryan Lowe et al. "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments". In: *Advances in Neural Information Processing Systems* 2017-December (June 2017), pp. 6380–6391. arXiv: 1706.02275. URL: http://arxiv.org/abs/1706.02275.

[95] Mostafa D. Awheda and Howard M. Schwartz. "The residual gradient FACL algorithm for differential games". In: *Canadian Conference on Electrical and Computer Engineering*. Vol. 2015-June. June. Institute of Electrical and Electronics Engineers Inc., June 2015, pp. 1006–1011. DOI: 10.1109/CCECE.2015.7129412.

[96] Sameh F. Desouky and Howard M. Schwartz. "Q(λ)-learning adaptive fuzzy logic controllers for pursuit-evasion differential games". In: *International Journal of Adaptive Control and Signal Processing* 25.10 (Oct. 2011), pp. 910–927. ISSN: 08906327. DOI: 10.1002/acs.1249. URL: http://doi.wiley.com/10.1002/acs.1249.

[97] Lionel Jouffe. "Fuzzy inference system learning by reinforcement methods". In: *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* 28.3 (1998), pp. 338–355. ISSN: 10946977. DOI: 10.1109/5326.704563.

[98] David Silver et al. "Mastering the game of Go with deep neural networks and tree search". In: *Nature* 529.7587 (Jan. 2016), pp. 484–489. ISSN: 14764687. DOI: 10.1038/nature16961.

[99] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (Feb. 2015), pp. 529–533. ISSN: 14764687. DOI: 10.1038/nature14236.

[100] Sergey Levine et al. "End-to-End Training of Deep Visuomotor Policies". In: *Arxiv* (Apr. 2015), p. 6922. arXiv: 1504.00702. URL: http://arxiv.org/abs/1504.00702.

[101] Greg Brockman et al. "Openai gym". In: *arXiv preprint arXiv:1606.01540* (2016).

[102] Yoshua Bengio et al. "Curriculum learning". In: *Proceedings of the 26th annual international conference on machine learning*. 2009, pp. 41–48.

[103] Ashvin Nair et al. "Overcoming exploration in reinforcement learning with demonstrations". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 6292–6299.

[104] Lu Jiang et al. "Self-paced curriculum learning". In: *Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015.

[105] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. "Cooperative multi-agent control using deep reinforcement learning". In: *International Conference on Autonomous Agents and Multiagent Systems*. Springer. 2017, pp. 66–83.

[106] Atsushi Kamimura and Toru Ohira. *Group Chase and Escape: Fusion of Pursuits-Escapes and Collective Motions*. Springer, 2019.

[107] Alex Graves et al. *Automated Curriculum Learning for Neural Networks*. 2017. arXiv: 1704.03003 [cs.NE].

[108] Scott Fujimoto, Herke van Hoof, and David Meger. *Addressing Function Approximation Error in Actor-Critic Methods*. 2018. arXiv: 1802.09477 [cs.AI].

[109] Rémy Portelas et al. *Automatic Curriculum Learning For Deep RL: A Short Survey*. 2020. arXiv: 2003.04664 [cs.LG].

[110] *Bebop Autonomy ROS Driver for Parrot Bebop Drone (quadrocopter) 1.0 2.0*. http://https://bebop-autonomy.readthedocs.io/en/latest/. Accessed: 2020-10-01.

[111] *Object feature extraction shape descriptors*. http://ojskrede.github.io/inf4300/notes/week_05/. Accessed: 2020-09-20.

[112] *ROS (Robot Operating System) - Documentation*. http://wiki.ros.org/. Accessed: 2020-10-01.

[113] *ROS (Fl-AIR - Framework libre AIR*. https://devel.hds.utc.fr/software/flair. Accessed: 2020-10-01.

[114] Nicola Roberto Zema et al. "CUSCUS: An integrated simulation architecture for distributed networked control systems". In: *2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE. 2017, pp. 287–292.

[115] BL Ho and Rudolf E Kálmán. "Effective construction of linear state-variable models from input/output functions". In: *at-Automatisierungstechnik* 14.1-12 (1966), pp. 545–548.

[116] Lucas Vago Santana, Alexandre Santos Brandao, and Mario Sarcinelli-Filho. "Outdoor waypoint navigation with the AR.Drone quadrotor". In: *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015* (2015), pp. 303–311. DOI: 10.1109/ICUAS.2015.7152304.

[117] Lucas Vago Santana, Alexandre Santos Brandão, and Mário Sarcinelli-Filho. "An automatic flight control system for the AR. Drone quadrotor in outdoor environments". In: *2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*. IEEE. 2015, pp. 401–410.

[118] Ben Yun, Kemao Peng, and Ben M Chen. "Enhancement of GPS signals for automatic control of a UAV helicopter system". In: *2007 IEEE International Conference on Control and Automation*. IEEE. 2007, pp. 1185–1189.

[119] Mohinder S Grewal and Angus P Andrews. "Applications of Kalman filtering in aerospace 1960 to the present [historical perspectives]". In: *IEEE Control Systems Magazine* 30.3 (2010), pp. 69–78.

[120] David Orton Wheeler. "Relative Navigation of Micro Air Vehicles in GPS-Degraded Environments". In: (2017).

[121] Achim Hornbostel. "Propagation problems in satellite navigation". In: *URSI Radio Science Bulletin* 2009.329 (2009), pp. 21–30.

[122] Michele Ballerini et al. "Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study". In: *Proceedings of the national academy of sciences* 105.4 (2008), pp. 1232–1237.

[123] Eric W Justh and Perinkulam S Krishnaprasad. *A simple control law for UAV formation flying*. Tech. rep. MARYLAND UNIV COLLEGE PARK INST FOR SYSTEMS RESEARCH, 2002.

[124] Ermin Wei, Eric W Justh, and PS Krishnaprasad. "Pursuit and an evolutionary game". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 465.2105 (2009), pp. 1539–1559.

[125] Wonsang Yoo, Eun Yu, and Jaemin Jung. "Drone delivery: Factors affecting the public's attitude and intention to adopt". In: *Telematics and Informatics* 35.6 (2018), pp. 1687–1700.

[126] Gordon D Hoople and Austin Choi-Fitzpatrick. "Drones for Good: How to Bring Sociotechnical Thinking into the Classroom". In: *Synthesis Lectures on Engineers, Technology, and Society* 9.1 (2020), pp. i–148.

[127] Joseph John Beel. "Anti-UAV defense for ground forces and hypervelocity rocket lethality models". PhD thesis. Monterey, California. Naval Postgraduate School, 1992.

[128] Dinakar Peri. "Expanding Anti-UAVs Market to Counter Drone Technology". In: *CLAWS Journal Winter* (2015), pp. 152–158.

[129] Georgia Lykou, Dimitrios Moustakas, and Dimitris Gritzalis. "Defending Airports from UAS: A Survey on Cyber-Attacks and Counter-Drone Sensing Technologies". In: *Sensors* 20.12 (2020), p. 3537.

[130] *Theiss - UAV solutions)*. http://www.theissuav.com/counter-uas. Accessed: 2020-10-23.

[131] *Hertz - anti-drone systems)*. https://www.hertzsystems.com/. Accessed: 2020-10-27.

[132] *Master List of Drone Laws (Organized by State & Country))*. https://uavcoach.com/drone-laws/. Accessed: 2020-10-27.

[133] *Anduril*. https://www.anduril.com/work. Accessed: 2020-10-27.

[134] *Boreades: CIVILIAN DRONE NEUTRALIZER SYSTEM*. http://boreades.fr/. Accessed: 2020-10-27.

[135] Slavimir S Nikolić. "An innovative response to commercial UAV menace: Anti-UAV falconry". In: *Vojno delo* 69.4 (2017), pp. 146–167.

[136] *Turkish anti-drone technology rolled out against 'aerial threats'*. https://www.hurriyetdailynews.com/turkish-anti-drone-technology-rolled-out-against-aerial-threats-140908. Accessed: 2020-10-27.

[137] *Counter Drone Solutions & Technology*. https://counterdronesolutions.com.au/technology/counter-drone/. Accessed: 2020-10-27.

[138] Felipe Cucker and Steve Smale. "Emergent behavior in flocks". In: *IEEE Transactions on automatic control* 52.5 (2007), pp. 852–862.

[139] Eliseo Ferrante et al. "Flocking in stationary and non-stationary environments: a novel communication strategy for heading alignment". In: *International conference on parallel problem solving from nature*. Springer. 2010, pp. 331–340.

[140] Manuele Brambilla et al. "Swarm robotics: a review from the swarm engineering perspective". In: *Swarm Intelligence* 7.1 (2013), pp. 1–41.

[141] Kwang-Kyo Oh, Myoung-Chul Park, and Hyo-Sung Ahn. "A survey of multi-agent formation control". In: *Automatica* 53 (2015), pp. 424–440.

[142] Alex Kushleyev et al. "Towards a swarm of agile micro quadrotors". In: *Autonomous Robots* 35.4 (2013), pp. 287–300.

[143] Osamah Saif. "Reactive navigation of a fleet of drones in interaction". PhD thesis. 2016.

[144] *Intel Drone Light Show at The Olympics*. https://www.dandad.org/awards/professional/2018/media/27058/intel-drone-light-show-at-the-olympics/. Accessed: 2020-11-02.

[145] Milán Janosov et al. "Chasing strategies of a flock of drones". In: ().

[146] *Boids 3D*. https://vvvv.org/contribution/boids-3d. Accessed: 2020-11-02.

[147] *27 Incredible Underwater Pictures of Schooling Fish*. https://allthatsinteresting.com/schooling-fish. Accessed: 2020-11-02.

[148] Gábor Vásárhelyi et al. "Optimized flocking of autonomous drones in confined environments". In: *Science Robotics* 3.20 (2018).

[149] Jason Welsby, Chris Melhuish, and Coldharbour Lane. "Autonomous minimalist following in three dimensions: A study with small-scale dirigibles". In: *Proceedings of Towards Intelligent Mobile Robots Manchster* (2001).

[150] John F Keane and Stephen S Carr. "A brief history of early unmanned aircraft". In: *Johns Hopkins APL Technical Digest* 32.3 (2013), pp. 558–571.

[151] *Guided Missiles R-27T1 / R-27ET1*. http://eng.ktrv.ru/production/military_production/air-to-air_missiles/r-27t1_-_r-27et1.html. Accessed: 2020-11-02.

[152] Tamás Vicsek. "Closing in on evaders". In: *Nature* 466.7302 (2010), pp. 43–44.

[153] Atsushi Kamimura and Ohira Toru. "Effective construction of linear state-variable models from input/output functions". In: *New Journal of Physics* 12.5 (2010), p. 053013.

[154] *How do you catch a drone? With an even BIGGER drone and a giant net: Tokyo police reveal bizarre 'UAV catcher'*. https://www.dailymail.co.uk/sciencetech/article-3356746/How-catch-drone-BIGGER-drone-giant-net-Tokyo-police-reveal-bizarre-UAV-catcher.html. Accessed: 2020-11-06.

[155] David Isele et al. "Navigating occluded intersections with autonomous vehicles using deep reinforcement learning". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 2034–2039.

[156] Sergey Levine et al. "End-to-end training of deep visuomotor policies". In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 1334–1373.

[157] OpenAI: Marcin Andrychowicz et al. "Learning dexterous in-hand manipulation". In: *The International Journal of Robotics Research* 39.1 (2020), pp. 3–20.

[158] John Schulman et al. "Trust region policy optimization". In: *International conference on machine learning*. 2015, pp. 1889–1897.

[159] Robert Lee et al. “Learning Arbitrary-Goal Fabric Folding with One Hour of Real Robot Experience”. In: *arXiv preprint arXiv:2010.03209* (2020).

[160] J. Matas, Stephen James, and A. Davison. “Sim-to-Real Reinforcement Learning for Deformable Object Manipulation”. In: *CoRL*. 2018.

[161] J. Chen, B. Yuan, and M. Tomizuka. “Model-free Deep Reinforcement Learning for Urban Autonomous Driving”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. 2019, pp. 2765–2771.

[162] Brendan Tidd, Nicolas Hudson, and Akansel Cosgun. “Guided Curriculum Learning for Walking Over Complex Terrain”. In: *arXiv preprint arXiv:2010.03848* (2020).

[163] Mike Schuster and Kuldip K Paliwal. “Bidirectional recurrent neural networks”. In: *IEEE transactions on Signal Processing* (1997).

[164] Sudeshna Pal. “A Review of Target Pursuit Strategies in Aerial Species”. In: *Dynamic Systems and Control Conference*. Vol. 46186. American Society of Mechanical Engineers. 2014, V001T05A004.

[165] Rolif Lima and Debasish Ghose. “Target localization and pursuit by sensor-equipped UAVs using distance information”. In: *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2017, pp. 383–392.

[166] Xiufang Shi et al. “Anti-drone system with multiple surveillance technologies: Architecture, implementation, and challenges”. In: *IEEE Communications Magazine* 56.4 (2018), pp. 68–74.

[167] *Fotografo registra foto rara de uma murmuracao, apelidada de ‘uma em um milhao*. https://catiororeflexivo.com/fotografo-registra-foto-apelidada-uma-um-milhao/. Accessed: 2020-11-18.

[168] Iain D Couzin, Jens Krause, et al. “Self-organization and collective behavior in vertebrates”. In: *Advances in the Study of Behavior* 32.1 (2003), pp. 10–1016.

[169] Iain D Couzin et al. “Effective leadership and decision-making in animal groups on the move”. In: *Nature* 433.7025 (2005), pp. 513–516.

[170] G. Sanahuja, P. Castillo Garcia, and A. Sanchez. “Stabilization of n integrators in cascade with bounded input with experimental application to a VTOL laboratory system”. In: *Int. J. Robust Nonlinear Control* 20.10 (2010). dx.doi.org/10.1002/rnc.1494, pp. 1129–1139. URL: http://hal.archives-ouvertes.fr/hal-00448201/en/.

[171] Pedro Castillo-Garcia, Laura Elena Munoz Hernandez, and Pedro Garcia Gil. *Indoor navigation strategies for aerial autonomous systems*. Butterworth-Heinemann, 2016.

[172] Alberto Castillo and Pedro Garcia. "Predicting the future state of disturbed LTI systems: A solution based on high-order observers". In: *Automatica* 124 (2021), p. 109365.

Contents