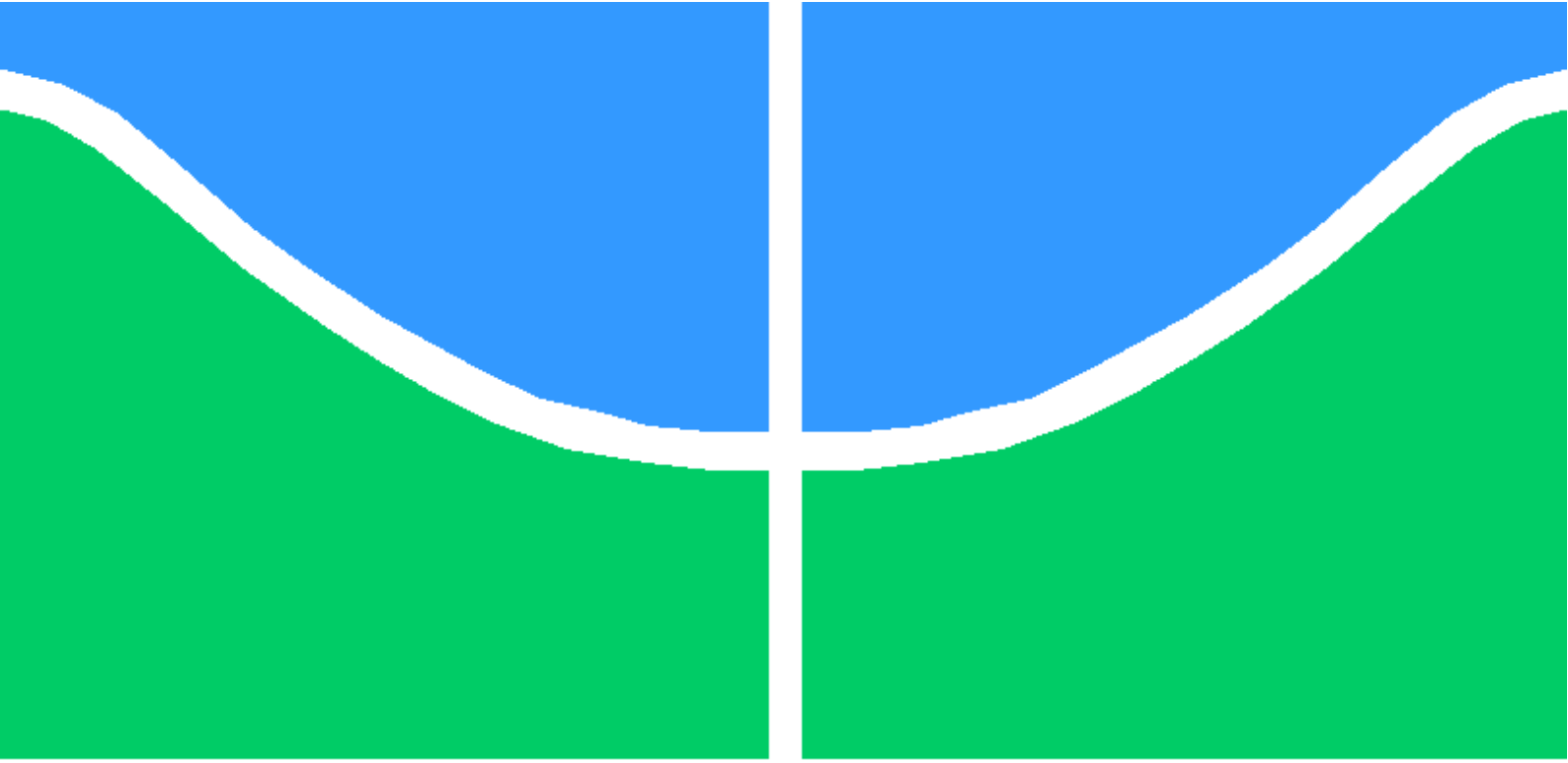


Master's Dissertation

Hybrid Control of a multi-agent UAV fleet for Formation Flight with Dec-POMDP

Bruno Rodolfo de Oliveira Floriano

Brasília, December 2019

UNIVERSITY OF BRASILIA

TECHNOLOGY FACULTY

UNIVERSITY OF BRASILIA
Technology Faculty

Master's Dissertation

Hybrid Control of a multi-agent UAV fleet for Formation Flight with Dec-POMDP

Bruno Rodolfo de Oliveira Floriano

Report submitted to the Electrical Engineering Department as parcial requirement to obtain the degree of Master in Electronics and Automation Engineering

Examination Board

Prof. Geovany Araújo Borges, ENE/UnB _____
Advisor

Prof. Henrique Cezar Ferreira, ENE/UnB _____
Advisor

Prof. João Yoshiyuki Ishihara, ENE/UnB _____
Internal examiner

Prof. Thiago Felippe Kurudez Cordeiro, FGA/UnB _____
External examiner

Prof. Renato Alves Borges, ENE/UnB _____
Alternate examiner

FICHA CATALOGRÁFICA

FLORIANO, BRUNO RODOLFO DE OLIVEIRA
Hybrid Control of a multi-agent UAV fleet for Formation Flight with Dec-POMDP
[Distrito Federal] 2019.
728/19, 52p., 210 x 297 mm (ENE/FT/UnB), Mestre, Dissertação de Mestrado - Universidade de Brasília. Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. Hybrid control
2. Formation flight
3. Dec-POMDP
4. UAV fleet

I. ENE/FT/UnB
II. Título

REFERÊNCIA BIBLIOGRÁFICA

FLORIANO, B. R. O. (2019). Hybrid Control of a multi-agent UAV fleet for Formation Flight with Dec-POMDP. Dissertação de Mestrado em Engenharia Elétrica, Publicação PGEA.DM - 728/19 Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 52p.

CESSÃO DE DIREITOS

AUTOR: Bruno Rodolfo de Oliveira Floriano
TÍTULO: Hybrid Control of a multi-agent UAV fleet for Formation Flight with Dec-POMDP
GRAU: Mestre ANO: 2019

———————————————-

Bruno Rodolfo de Oliveira Floriano
SQS 209 Bl C Ap 105, Asa Sul
70.272-030 Brasília - DF - Brasil

Dedication

Dedico este trabalho aos meus pais, Lúcia e Juscelino, ao meu irmão, Vinícius, e à minha namorada, Jéssica.

Bruno Rodolfo de Oliveira Floriano

Thanks

I would like to thank my family for all the support provided during the period of intense studies and work on this dissertation. Also to my girlfriend Jessica for the great support and comfort at all the difficult moments. To my advisors Geovany and Henrique who always provided me with good advises both technical and personal. Finally, to all my friends at the university for the moments of necessary distractions.

Bruno Rodolfo de Oliveira Floriano

RESUMO

Voo em formação e controle cooperativo de múltplos VANTs têm sido áreas de estudo de grande interesse das pesquisas mais recentes. Enquanto diversos métodos estão sendo criados para rastreamento fino de referência e formação, muitos empecilhos ainda precisam ser superados tais como descentralização, comunicação confiável, divisão de tarefas, evitamento de colisões e autonomia. Neste cenário, este trabalho propõe um sistema de controle híbrido para ser usado no voo em formação de múltiplos VANTs de asa-fixa, aumentando a performance e eficiência do grupo por permitir que este planeje e controle a frota através de comandos discretos e contínuos. Para contornar o problema da centralização, o método de planejamento Dec-POMDP foi utilizado, de modo a evitar a confiabilidade em um nó central de tomada de decisão, como um líder ou uma estação em terra. Através do uso deste algoritmo, este método também considera transições e observações estocásticas para permitir uma tomada de decisão eficiente mesmo em ambientes ruidosos e incertos. Além disso, a implementação deste sistema em uma malha externa permite reduzir o tempo computacional. Através de simulações, o sistema proposto como uma topologia chaveada entre a política Dec-POMDP e controles PID foi comparada com outros métodos da literatura e apresentou uma performance satisfatória para o voo em formação.

ABSTRACT

Formation flight and cooperative control of multiple UAVs has been areas of studies of great interest by the most recent researches. As many methods are being created to make fine reference and formation tracking, collision avoidance and disturbance rejection, many trammels are still necessary to be overcome such as decentralization, reliable communications, task division, obstacle avoidance and autonomy. In such scenario, this work proposes an hybrid control system to be used in formation flight of multiple fixed-wing UAVs, increasing the group performance and efficiency by allowing it to plan and control the fleet by using both discrete and continuous commands. To overcome the centralization problem, the Dec-POMDP planning method is used, in order to avoid the reliability on a central decision node, such as a leader or a ground station. By using such algorithm, this approach also considers stochastic transitions and observations to allow an effective decision making in noisy and uncertain environments. Also, the implementation of such system in an outer loop allows to reduce the computational time. Through simulations, the system proposed as a switching topology between the Dec-POMDP policy and PID controls was compared to other methods in the literature and has presented satisfactory performance for formation flight.

SUMMARY

LIST OF FIGURES

LIST OF TABLES

LIST OF SIMBOLS

Latin Symbols

C	Adjacency Matrix	
c	Communication weight of a pair of vehicles	
s	Global state	
t	Time	[s]
V	Lyapunov artificial potential function	
x	Vehicle's three-dimensional position	[m]
u	Control input	[m/s]
S	Set of global states	
b	Probability distribution of states	
A	Set of possible actions	
a	Action	
T	State transition probability function	
R	Reward function	
Z	Set of possible observations	
z	Observation	
O	Observation probability function	
h	Time horizon	
γ	Discount factor	
E	Expected value	
I	Set of agents	
n	Position white noite	[m]
v	Velocity	[m/s]
d	Altitude	[m]
f	Desired formation	[m]
g	Navigation action function	
K	PID parameter	
D	PID transfer function	
N	Filter coefficient	
e	Error	[m]

Greek Symbols

ν	Set of nodes in a graph	
ξ	Set of edges in a graph	
∇	Gradient operand	
π	Policy	
τ	Normalizing factor	
ψ	Heading angle	[deg]
ω	Angular velocity	[rad/s]
ε	Distance constants	[m]
δ	Aerodynamic controls	
α	Acceleration	
β	Normalizing factor	
λ	Distance scalar factor	[m]
ρ	Distance scalar factor	[m]
σ	Position standard deviation	[m]

Dimensionless Groups

N	Total number of UAVs in the fleet

Subscripts

i	i-th agent of the group
j	j-th agent of the group
n	n-th agent of the group
ij	coupling between the i-th and the j-th agent
0	Initial time
t	Current time / Throttle
r	Reference
d	Distance / Altitude
p	Proximity margin
c	Communication margin
v	Linear
ω	Angular
b	Base value
a	Aileron
e	Elevator / Error
P	Proportional
I	Integral
D	Derivative
f	Filter
z	Altitude

Superscripts

$\dot{\cdot}$	Temporal variation
$-$	Target value
$*$	Optimized
$'$	Next time step / Alternative
$\hat{}$	Measured
i	i-th agent of the group

Acronyms

UAV	Unmanned Aerial Vehicle
UnB	University of Brasilia
IMU	Inertial Measurement Unit
CPU	Central Processing Unit
GPS	Global Positioning System
GCS	Ground Control Station
PID	Proportional-Integral-Derivative controller
POMDP	Partially Observable Markov Decision Process
Dec-POMDP	Decentralized Partially Observable Markov Decision Process
I-POMDP	Independent Partially Observable Markov Decision Process
MPOMDP	Multi-agent Partially Observable Markov Decision Process

Chapter 1

Introduction

1.1 Contextualization

Thanks to the great advances in electronic researches in the last few decades, there's been a fast and continuous increase in computational capacity, which consequently allowed for the development of powerful embedded technologies [1, 5, 6]. Such improvements have made possible to create complex and robust control algorithms that can be used in real scenarios, bringing a high number of possibilities of usage. One of the major applications to benefit from such progress, is the development of flying vehicles such as unmanned aerial vehicles (UAV).

UAV is a type of aerial vehicle, such as a fixed-wing aircraft (airplane), a quad-rotor, an helicopter or any other vehicle that can fly, that doesn't require a human pilot on board. For an effective flight, this kind of system requires well constructed algorithms that deal with all parts of the vehicle's control, such as the path planning, the non-linear aerodynamics and the obstacle avoidance. If well built, such systems can be quite versatile, and thus might be used in many different applications including military, video and photography, package delivery, surveillance, target tracking, traffic monitoring and many others [1, 7, 8].

Those systems are now not only trustworthy but also commercially available, thanks to the great technology advance. Indeed a great number of these examples can be found today as companies, governments, civilians and researches are building or using such vehicles in their daily routines. The company *Amazon.com, Inc.*, e.g. made their first air delivery in December 7th, 2016 [1]with the quad-rotor shown in Fig. 1.1 . Not to mention the numerous models of quad-rotors available for video and photography at a wide range of prices, such as the ones sold by the Chinese company *DJI*. Figure 1.2 shows the *Mavic Air*, one of their products [2].

As each individual vehicle becomes more advanced and less costly, the scientific research looks for novel ways of extrapolating such technology in order to obtain the better and most efficient use of it. As for the unmanned aerial vehicles (UAVs), the flight of multiple units is one the most notable enhancement that is being researched [5, 8].

[1] https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011
[2] https://www.dji.com/br/mavic-air?site=brandsite&from=nav

Figure 1.1: Amazon's quad-rotor for package delivery (Source: [3])

Flying more than one UAV at the same time may have many advantages with respect to only a single vehicle. For example, the area that can be covered with a numerous fleet is significantly larger than what can be detected with a lonely agent. This allows for a larger and more accurate visual recognition, mapping or target detection, which can be used, e.g. for geographical research, data collection, surveillance and etc [3].

Another possibility that comes with a multi-agent group is task differentiation [9]. Some situations require that many tasks are performed in order to accomplish a given goal, which can be processed inefficiently, or even impossible to be performed, by a single unit. On the other hand, a group of UAVs can be separated with respect to tasks and therefore perform all of the mission requirements. For example, task differentiation can be used in rescue missions: one vehicle can search for humans that need assistance while another one acts to aid each person detected by the first one.

Finally, even if the goal doesn't require different tasks, a multi-agent fleet can add redundancy to a given mission, such that a vehicle can substitute another in the case of a downfall or a communication lost. This is useful and necessary in military scenarios, for example [8].

For those goals and advantages to be fully harnessed in a formation flight, one should consider several different objectives in the flight planning, including the tracking of a global and known reference, the desired geometric pattern of the formation, the collision avoidance between vehicles or with an external obstacle, the communication and sensors noise robustness and many other factors [1]. To perform such tasks, it's necessary to have efficiency in both control and planning algorithms.

[3] https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011

Figure 1.2: DJI's quad-rotor called Mavic Air (Source: [4])

1.2 Motivation

The Aerial Robotics Laboratory is a facility of the University of Brasilia (UnB) responsible for many researches, developments and implementations related to UAVs. A picture of the laboratory is show in Fig. 1.3. In the lab, researchers and students work with unmanned helicopters, quad-rotors, articulated bi-rotors and, mainly, fixed-wings aircraft.

There are a total of 9 different fixed wing UAVs of 4 distinct models. The main platform that is currently being developed is based on the *Volantex RC Ranger EX* model [5], shown in Fig. 1.4. There are a total of 5 units of this plane, in which 3 are available for the formation flight project.

The embedded system that will be on board of these planes and its interconnection can be visualized in Fig. 1.5. It is composed mainly by a digital autopilot *Pixhawk 2* (with a built in inertial measurement unit - IMU), a central processing unit (CPU) *Gumstix Overo*, a modem for communications among agents *microhard Pico Series P900* and a precise GPS system *Here+ RTK GPS*.

The autopilot has several specific commands that allows that a regular flight may already be performed even without any external processing unit. There are many different forms to program a specific flight, with the aid of a Ground Control Station (GCS) provided with a suited flight software, such as *Mission Planner* [6]. However, due to the scope of the project and the final aim in getting a whole fleet to flight in formation, an external CPU is required to better deal with communications with other vehicles, as well as to compute more dynamic and complex control functions.

The connections of the modules are built as follows: the communication modem change data

[4] https://www.dji.com/br/mavic-air?site=brandsite&from=nav

[5] https://www.volantexrc.eu/volantex-rc-ranger-ex-long-range-fpv-uav-platform-unibody-big-weight-carrier-v757-3-pnp-p-224.html

[6] http://ardupilot.org/ardupilot/index.html

Figure 1.3: UnB's Aerial Robotics Laboratory

with other vehicles with the same device. Those data are sent to (or taken from) the CPU which processes those information. As a given command is necessary for the flight algorithm within the CPU, it is sent to the autopilot, which then performs it by sending a response to the motors (throttle, aileron, rudder or elevator). As the plane change its position and attitude, the sensors (accelerometer, gyroscope and etc.) and GPS update their status, sending them to the autopilot. The autopilot, in its turn, will close the loop by sending the UAV data to the CPU to process. A human pilot has access to the autopilot through a specif radio controller system in case some emergency command should be performed.

Hence, the Aerial Robotics Laboratory has appropriate equipment and structure to build a real system of multiple fixed-wings UAVs to flight in formation and test in practice the algorithms for planning and control.

1.3 Objectives

This work's objective is to develop a novel outer loop hybrid control system to be used in a multiple fixed-wing UAV fleet in order to achieve a decentralized formation flight. By adopting the hybrid approach, the system might be benefited from the fine reference tracking made possible by a continuous controller such as a PID set, and also have an efficient decentralized decision making in unusual situations by a Dec-POMDP policy, even in a noisy environment and without any collisions. The use of high-level actions can decrease the number of states and, therefore, improve the computational time of the optimization solving. Similar to [10], this system might take the advantages that come with both continuous and discrete controllers and, therefore, be most efficient than an individual method.

Ultimately, this algorithm should be able to appropriately track a global reference in a given formation structure also taking into account an uncertain environment with noisy communications as well as considering collision avoidance. Simulations should be performed in order to test the

Figure 1.4: Fixed-Wing UAV Ranger EX (Source: [7])

system to validate its performance and compare it to other approaches found in literature. The final goal is to implement this control system in the real fleet of the Aerial Robotics Laboratory to perform a real flight.

1.4 Results

Through simulations performed, the proposed hybrid system was validated and compared to other methods in the literature. To test the efficiency of this method in different UAVs, two distinct models were used with our algorithm, showing results that are comparable to the existing algorithms in the literature. Furthermore, the system was tested with distinct references and formations and managed to appropriately track them.

Due to the lack of a theoretical model of the real UAVs, this system was still not able to be implemented in practice. However, it was properly built to fit different models and, as such, the real flight is a suggestion for future works once the model definition is concluded.

1.5 Dissertation organization

This dissertation is organized as follows: Chapter 2 shows a literature review of the most recent works on formation flight, specially with decision-making algorithms such as POMDP and its extensions. Chapter 3 describes the methodology used to develop the proposed algorithm. Simulated results are discussed in Chapter 4, followed by the conclusion remarks in Chapter 5. The appendix contain complementary materials.

[7] https://www.volantexrc.eu/volantex-rc-ranger-ex-long-range-fpv-uav-platform-unibody-big-weight-carrier-v757-3-pnp-p-224.html

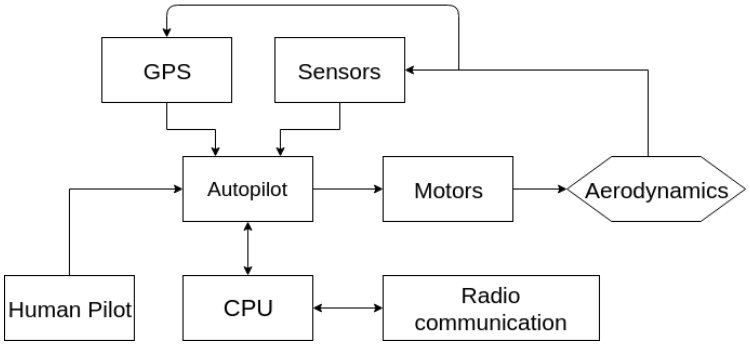


Figure 1.5: Embedded System of the plane

Chapter 2

Literature Review

2.1 Introduction

In recent years, many researches in the field of aerial robotics have been conducted in order to built different control techniques and planning algorithms to accomplish an effective flight that might suit one of the many applications described in Chapter 1. Those include, for example, consensus and synchronization methods to deal with communications problem such as noise, stochastic information and dynamic topologies; formation strategies such as leader-follower, virtual structure and behavior based to account for the desired group management and reference tracking; and optimization and estimation systems such as Kalman filters, to deal with the better performance for a given application [5, 11].

2.2 Consensus and synchronization

Flying in formation with multiple UAV units requires a consistent knowledge of the world states, as well as the information about the other vehicles [1]. Knowing with precision the data that surrounds a given agent, as well as its own internal data, allows the embedded control system to act accordingly and consequently it can refine the control output, avoid the collision with other agents, diversify the tasks allocation and many other possibilities depending on the mission requirements.

For those reasons the consistency of the information between the vehicles in the group is of great importance in formation flight and other types of missions with multiple units. Therefore, the agents should be able to update the status of the states it sees by communicating with its counterparts. That is the main concern for the consensus and synchronization techniques.

However, communication is not often available or robust enough such that all the vehicles have the precise data from all the other ones in the group. Indeed, those data might be corrupted by noise and disturbances which is not unusual in flying scenarios due to atmospheric noise and limited embedded communication systems. For that reason a consensus algorithm deals with the

most effective way of converging a given state, in the perception of all the agents, by exchanging information with the vehicle's neighbor.

2.2.1 Graph Theory

One of the main models to represent the agent's communication topology is by using the graph theory [5]. As it depicted in Fig. 2.1, a directed graph is defined by a set of nodes, $\nu = \{1, \ldots, N\}$, representing the N agents in the group, and by a set of edges, $\xi \subset \nu \times \nu$, composed by the combinations of two nodes representing the communications between two vehicles. In this definition it is possible that a vehicle receives data from another but the opposite case might not be true, that is why it is defined as a directed graph. On the other hand, an undirected graph is a special case of the previous in which each edge represents the mutual transmission and reception between two nodes.

Furthermore, it is also possible to weight each edge by giving it a particular value. It might be done through the adjacency matrix $C = [c_{ij}] \in \Re^{N \times N}$ such that the element c_{ij} weights the relevance of the communication that vehicle j sends to i. This allows the system to account for a particular connection to be stronger (i.e. more reliable) than the others, to be weaker or even nonexistent ($c_{ij} = 0$).

2.2.2 Consensus Algorithm

Usually, the consensus algorithm used to update a given state, $s(t)$, is modeled through a differential equation that imposes the dynamics of the synchronization law of the vehicles' communication [1]. The most used equation to represent that dynamic is

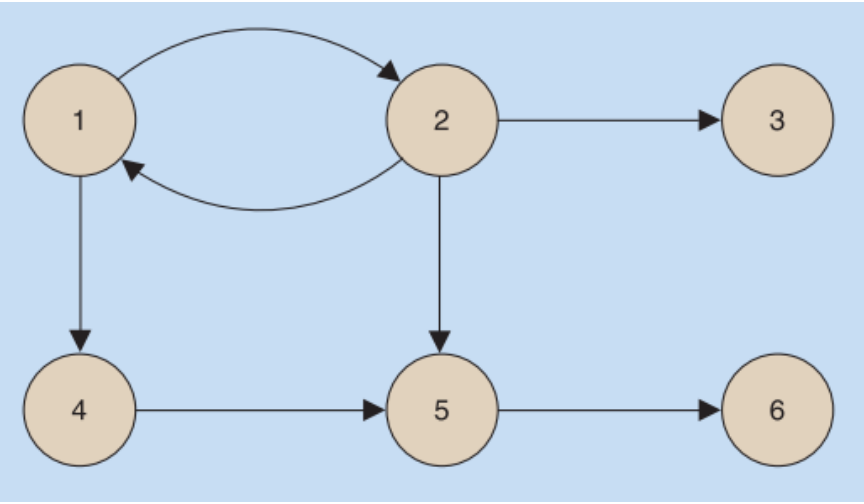


Figure 2.1: Communication directed graph (Source: [1])

$$\dot{s}_i(t) = -\sum_{j=1}^{N} c_{ij}(t)(s_i(t) - s_j(t)), \quad i = 1, \dots N, \tag{2.1}$$

in which, $s_i(t)$ is the information state as seen by the i-th vehicle at time t and $c_{ij}(t)$ is the corresponding (i, j) element of the adjacency matrix at time t. By assuming this element to be time-varying, one includes the case in which the communication topology of the group is also time-varying and, therefore it can change in a single flight due to a wide maneuver, a communication loss and etc. The Eq. 2.1 also shows that the information update of each vehicle tends to follow the data of its neighbors (weighted for each communication link). To model a similar update law for discrete signals, a difference equation should be used instead [1].

The consensus and synchronizations methods extend beyond that, studying the conditions for convergence or also including analysis for time delays in the communication, stochastic data, switching topologies, asynchronous updates and many other cases [1, 12].

2.3 Formation control

After dealing with the communication system through a consensus algorithm, the formation control should be designed in order to achieve an effective formation flight. Indeed the system that manages to take the whole group to a given formation, either a static or dynamic one, while tracking a global reference or avoiding obstacle, is one the most challenge to be studied and implemented as it includes many characteristics.

The main objective of such system can be resumed in taking all the N agents of a group of vehicles (in this work, UAVs), to form a particular geometric pattern, which can be either fixed or time-varying. Assume, for example, a group of $N = 4$ agents, as depicted in Fig. 2.2, represented by the black circles. The formation objective in this case is to take all of the vehicles to form the geometric pattern of a diamond. The reference point of the formation might be chosen depending on the situation: it may be one of the agents (a leader for instance) or a global reference. In this case, it is the central cross that marks the center of the diamond. Therefore, the formation control strategy should be able to take each individual to its particular position in the formation, here represented as the white circles. Also depends on the strategy (and the mission requirements) to determine if any vehicle is able to fill any vertex, or if each one has its own particular goal.

Although that is the primary goal of a formation flight control system, it should be noted that there are several other objectives that are of equal importance such as to avoid the obstacles that might be on the route of each agent, to avoid the collision among vehicles, not to mention a specific mission objective such as target tracking, area coverage, data collection and etc.

To accomplish those goals, several methods have been developed, simulated and tested in the last decades. The main classifications of such algorithms can be divided into the leader-follower strategy, the behavior based method and the virtual structure based approach [13].

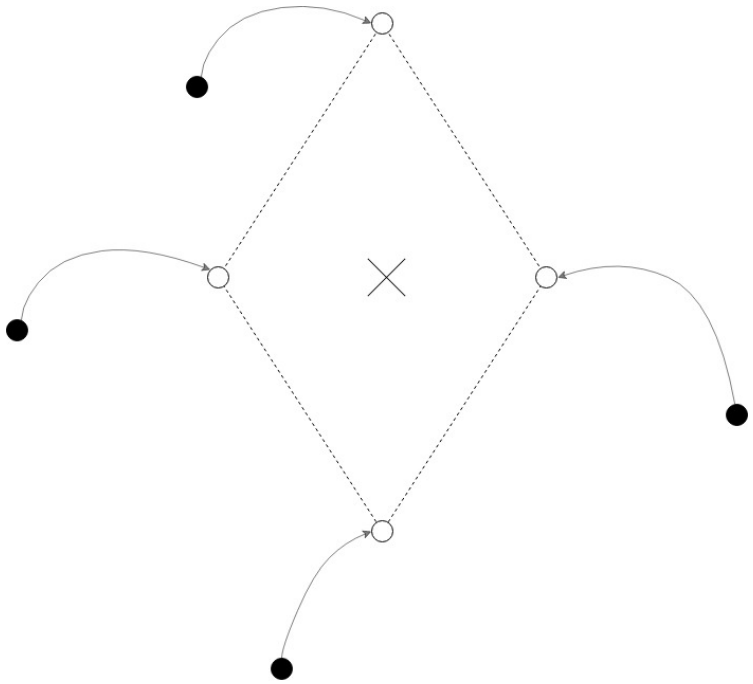


Figure 2.2: Formation desired as a geometric pattern

2.3.1 Leader-follower strategy

The leader-follower strategy is one of the most used techniques to control a given group of vehicles (not only UAVs) [8]. Due to its easy implementation this approach has been the most popular to be studied in the multi-vehicle formation problem. However, this method has some disadvantages such as the communication reliability and redundancy issues, which might be mitigated by some of the leader-follower variations.

The main principle of the this strategy is that, from the N agents that compose the group, one of them is designated to be the leader. It will be the one to receive the global reference to be followed. The other agents, designated as followers, will have to communicate with the leader to receive its position and then track a specific distance from it.

The easiness of such strategy relies on the fact that only one agent must possess the main information (the global reference) and the others should only follow it by receiving the data from the communication system. It makes the system not only simple to be implemented but also scalable, once the addition of more agents requires only the direct knowledge of the leader status [14].

However, such ease comes with some drawbacks. The first one is the communication reliability: the followers have a high dependency on the information that is been transmitted from the leader. If there is some failure in the communication system, or even noise at the transmission channel, the only reference of the followers is corrupted and then they might get lost. This is specifically important in the UAV case, once the environment in which the agents will be acting is very noisy and uncertain.

Another possibility to take into account is the redundancy: consider the case in which the leader is somehow lost (the vehicle was turned off, shot at or simply stopped responding). If this is the case, similar to what was previously mentioned, the followers are now lost, once the only reference they have is not sending the data they need to keep track of its tasks. Therefore, the group lacks of redundancy.

To take those problems into consideration some extensions of the classic leader-follower method have been developed. The first one is the virtual leader in which the leader is not one of the N agents, but a common virtual reference [7, 15]. In this approach, all of the vehicles have a global position knowledge that represent a reference point to guide them in their flight. By adopting such method, the fleet can minimize the communication dependency and also can improve its redundancy robustness.

The virtual leader is the main variation of the leader-follower method due to the balance between ease of implementation and communication and redundancy robustness. For those reasons it was chosen for this work.

There were still a few other extensions of the leader-follower strategy such as the implicit leader [8]. In the method proposed by He et al., there is no explicit leader, i.e. the followers don't have the information of which agent is a leader, whereas it only relies on the proposed consensus algorithm that models the information exchange between vehicles to know the reference. Also, in this method the leaders are dynamic, which means that each vehicle can be a leader depending on the situation. By itself, the dynamic leader method is another strategy in the leader-follower scope of algorithms.

2.3.2 Behavior based method

The behavior based method consists of assigning each robot several tasks (or behaviors) to be accomplished [7]. Then, the group general performance will be a weighted result of all the behaviors contributions. This method is particularly interesting for its ability to accomplish multiple tasks at the same time (which might be desirable depending on the mission goals) as well as its possibility to respond to fast environmental change [16].

However, many disadvantages make this strategy undesirable to be implemented in many cases. The first inconvenience is that it requires individual predefined behaviors [16]. This condition weakens the scalability of such systems, once each new individual should be assigned a specific behavior. In addition, the individually of such behaviors makes it difficult for the formation to achieve stability. Indeed the converge itself of the group state may be hard to be mathematically guaranteed [7].

As it will be seen further, by using decentralized decision making algorithms such as the Dec-POMDP, a certain degree of individual behavior might be added to a flying system (therefore, maintaining some of the advantages of the behavior based method) while still using a virtual leader and followers approach (and also keeping scalability and improving the stabilization).

2.3.3 Virtual structure based approach

In the virtual structure based approach, the whole group is modeled as a single entity, in which the tasks and behaviors rely on [7]. Therefore, each individual is designed to keep coordination in respect to the group task. This method is better suited for scenarios in which the group is determined to follow a very close and unchangeable structure.

One the other hand, the rigidity in the formation has the inconvenience of the lack of flexibility. Thus, if there is an obstacle of some sort in the environment, or if the formation is somehow required to change, than this strategy may not guarantee the flexibility to do so. For those reasons the virtual structure approach is not usually chosen to guide groups of UAVs.

2.3.4 Lyapunov Artificial Potential Function

While those methods dictate the relations between each individual and its role in the group, a more specific control technique is required in order to actually drive the agents to their relative position in the formation, accordingly to what was previously stated to be the concrete goal of the formation control. One of the most used methods to do that, which might be implemented alongside one the previous strategies, is the Lyapunov artificial potential function [17].

A potential function, either artificially created or not, is a function that weights the capability of a system, given a specific state, to generate or retain energy. For instance, the gravitational potential energy function weights the magnitude of energy that an object has given its current altitude. As it drops, and consequently loses height, so as the potential function drops until it gets to its minimum value.

The artificial potential function, in its turn, tries to mimic the natural behavior of those kind of energy functions in order to achieve a certain goal by trying to reach the point of minimum potential. Then, let $V(x_1, \ldots, x_N)$ be an artificial potential function of the vehicles' positions, x_i. It should be artificially designed such that it achieves its minimum value when all of the agents are at their desired points in the formation. If the target distance between two agents i and j is given by $\bar{x}_{ij}$, then the Lyapunov artificial potential function can be implemented as

$$V(x_i, \ldots, x_N) = \frac{1}{4} \sum_{\{i,j\}} \left(||x_i - x_j||^2 - \bar{x}_{ij}^2 \right), \tag{2.2}$$

such that, if $||x_i - x_j|| = \bar{x}_{ij}$ for all i and j, than the function V will be equal to zero, the minimum possible value. The absolute values within Eq. 2.2, associate with the squares operands, make sure that the function is always non-negative which leads to the possibility of stabilization.

The potential function stated in Eq. 2.2 models the desired behavior of the vehicles with respect to the target formation to be achieved. However, in order to actually drive the group towards the specific geometric pattern the vehicles should be controlled through a function that actually leads them to the point of minimum potential. To achieve that it is determined that the control input u_i of the i-th agent is obtained through

$$u_i = \dot{x}_i = -\nabla_{x_i} V(x_i, \ldots, x_N), \tag{2.3}$$

in which the gradient operand ∇_{x_i} is done with respect to each agent position x_i. The idea behind this operation is that it creates a vector field that, when added the negative sign, points in the direction of the decreasing potential function. This way the input of each vehicle will drive its position to the point of minimum artificial energy which was modeled to be exactly at the target formation.

By using a model that implements a similar artificial potential function, the group can be driven to the target position just as a natural system seeks for the point of minimum energy consumption. Indeed this idea can be implemented even with obstacles, i.e. the system sees a detected obstacle as a point of very high (possibly infinite) potential which then should be avoided (or by using a positive gradient in order to create a vector that points away from the obstacle).

While very robust and refined, the artificial potential function method still has some issues. The first one is that the stabilization in the point of global minimum can not be guaranteed, once the implementation might trap the system in a local minima [7]. This leads to a converge problem that might be difficult to overcome. Another issue with this method is the communication required in order to perceive the other agents around the vehicle to keep the correct distance between all of them. This implementation might be problematic in the aerial environment. Also, this approach resemble the virtual structure based method in the sense that it doesn't explicit considers sudden formation changes, vehicle redundancy or different tasks.

2.4 POMDP

Markov decision process (MPD) is a planning method developed to crate polices that directly relate discrete states to discrete actions and, therefore can make a system (such as a robot) to interact with the world in an optimized way according to a Markov transitioning model [2].

The MDP algorithms not only relate states with actions, but it is also able to keep track of the history of actions and observations it has made and use it to refine its knowledge of the world. A reward system also manages to determine which situation is more desirable for that specific system and can use it alongside with the Markov transitions to better estimate the impact of the present action in the future rewards.

However, the MDP by itself does not take into account the inaccuracy in which a real system (with real and noisy sensors) perceives the world states. Because of that an extension to MDP model was developed to consider this uncertainty and was called Partially Observable Markov Decision Process (or POMDP).

In the POMDP model the action is not directly determined through the states, but through the discrete observations that the system have of the states. Due to the uncertainty that lies on the sensors and the world itself, each observation is a probabilistic function of the states. Therefore, the decision is made based on the probabilities of each state in that give moment.

Formally, the POMDP method can be described by using the following tuple $\{S, A, T, R, Z, O, h, \gamma\}$, in which,

- S is the set of global states with initial distribution b_0;
- A is the set of possible actions;
- T is the probability function of transitioning from state $s \in S$ to state $s' \in S$ by taking the action $a \in A$, i.e. $T(s, a, s') = P(s'|a, s)$;
- R is the reward function of being in state $s \in S$ and taking the action $a \in A$, i.e. $R : S \times A \rightarrow \mathbb{R}$;
- Z is the set of possible observations of the agent;
- O is the probability function to observe $z \in Z$ after taking the action $a \in A$ that results in state $s' \in S$, i.e. $O(z, a, s') = P(z|a, s')$;
- h is the horizon of the problem, i.e. the number of time steps until termination.
- γ is the discount factor that weights the relevance of future rewards in comparison to the present.

The POMDP goal is to decide which action (from the set A) should be chosen at each time step, from the beginning of the task until the horizon h, in order that the expected reward of the system is maximized. In order to achieve that, the POMDP solver should build a policy tree, as seen in Fig. 2.3. It is a graphic representation of the policy, π, i.e. the function that outputs the action that should be taken at each time step given the previous observation made. The policy that gives the maximum cumulative reward is represented as π^* and should be found through

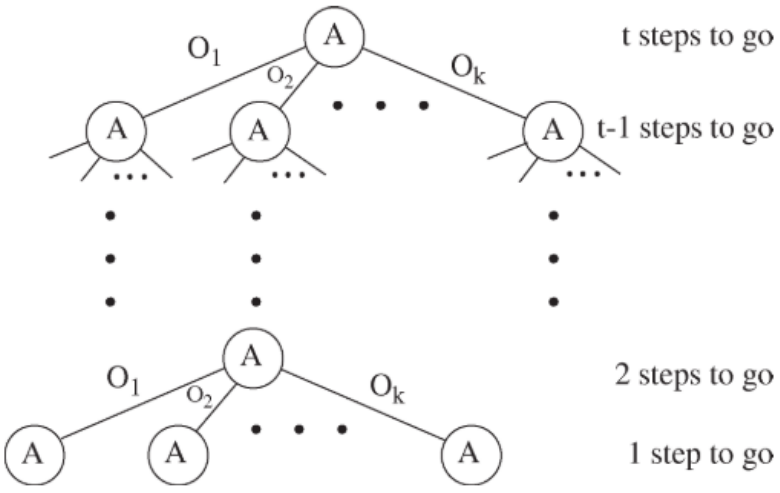


Figure 2.3: Policy Tree of POMDP (Source: [2])

$$\pi^* = \arg\max_{\pi} E\left[\sum_{t=0}^{h-1} \gamma^t R(a_t, s_t)|s, \pi\right]. \tag{2.4}$$

Equation 2.4 states that the optimized policy, π^*, is given as the argument which maximizes the expected cumulative reward, from time $t = 0$ until termination at the horizon. The reward is weighted by the discount factor γ in order to make the present decisions more relevant than the future ones. Therefore, solving Eq. 2.4, gives a direct instruction of what the agent should do given its past observations and the current time step.

As the policy tree is constructed during the decision making process, it is given a certain value $V^{\pi}(s)$ for each state s. This value is a direct measurement of the reward of such policy, however accounted for the future rewards yet to be received. Therefore it can be written as

$$V^{\pi}(s) = R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') \sum_{z \in Z} O(s', a, z) V^{\pi'}(s'). \tag{2.5}$$

Equation 2.5 can be interpreted as follows: the value of a given policy π at state s is given by the reward of being in that state, following the action a (decided by the policy), added by the value to be received in the following time step. This term, on the other hand, have to acknowledge the probability that current state s will be followed by any possible s', as well as the possibility that the agent might make any observation within the set O and not the state itself.

Finally, given the current belief state $b(s)$ (i.e. the probability distribution of being in state s) than the policy's value can be found through

$$V^{\pi} = \sum_{s \in S} b(s) V^{\pi}(s). \tag{2.6}$$

To find the optimum policy π^*, it is only necessary to find which policy, π gives the greater value. The evolution of the belief state at each time step is given by

$$b'(s') = \tau O(s', a, o) \sum_{s \in S} T(s, a, s') b(s), \tag{2.7}$$

in which τ is a normalizing factor to make the vector of the next belief state, $b'(s')$, sum up to one.

2.5 Dec-POMDP

The Decentralized POMDP (Dec-POMDP) framework is a planning method for multi-agent problems, designed in such way that the observation-action policy obtained is decentralized, which avoids the necessity of adding a central processing unit in which the vehicles might depend on. To model a problem in a Dec-POMDP frame, it should be described as global states and rewards, individual actions and observations and stochastic transition and observation functions, such as its parent method POMDP [18].

It can be formally defined by the tuple $\{I, S, \{A_i\}, T, R, \{Z_i\}, O, h, \gamma\}$, in which [19]:

- I is the set of agents, from 1 to N;
- S is the set of global states with initial distribution b_0;
- A_i is the set of individual actions for the i-th agent ($i \in I$). In addition, $A = A_1 \times \cdots \times A_N$ is the joint action of the group, in which $\times$ is the Cartesian product operator;
- T is the probability function of transitioning from state $s \in S$ to state $s' \in S$ by taking the joint action $a \in A$, i.e. $T(s, a, s') = P(s'|a, s)$;
- R is the global reward function of being in state $s \in S$ and taking the joint action $a \in A$, i.e. $R : S \times A \to \mathbb{R}$;
- Z_i is the set of individual observations of the i-th agent ($i \in I$). In addition, $Z = Z_1 \times \cdots \times Z_N$ is the joint observation of the group;
- O is the probability function of jointly observe $z \in Z$ after taking the joint action $a \in A$ that results in state $s' \in S$, i.e. $O(z, a, s') = P(z|a, s')$;
- h is the horizon of the problem, i.e. the number of time steps until termination.
- γ is the discount factor that weights the relevance of future rewards in comparison to the present

Let the individual policy, π_i be a function that maps the observations made by the i-th agent, z_i, to a given action, a_i. Then, the global policy of the group as a whole can be defined as $\pi = \pi_1 \times \cdots \times \pi_N$. The main objective of the Dec-POMDP algorithm is to find the optimized global policy, π^*, i.e. the function that results in the maximum expected cumulative reward. Mathematically,

$$\pi^* = \arg\max_{\pi} E\left[\sum_{t=0}^{h-1} \gamma^t R(a_t, s_t)|s, \pi\right]. \tag{2.8}$$

This optimization problem is built to be resolved in a offline planning phase by some of the many existent search algorithms [18]. After the optimized policy is achieved and, consequently, the individual policies, then they can be added to the real system to be executed in an online phase. Although the optimization is done by considering a global reward, the decentralization is still valid once each agent receives its own policy, thus it should act based only on its own observation.

2.5.1 Other mutli-agent POMDP variations

While the Dec-POMDP is a suited model for planning in a multi-agent scenario while still maintaining decentralization, it should be noted that there are many other POMDP variations that may deal, in different ways, with the multi-agent case. The work of Capitan et al. [3] gives a

brief presentation about of the POMDP variations that deals with multiple agents. It classifies each different method based on both the communication dependence among agents and the coupling level of the resulting policies. This classification is better depicted in Fig. 2.4.

The first and simpler approach is the independent POMDPs (I-POMDP). This strategy considers that each agent in the group has its own POMDP policy to be solved individually, never considering the existence of the its other counterparts. Due to this simple and disconnected approach, this method is classified as the least dependent on communication and with less coupling between agents. Although easy to be implemented, the disconnection and independence among vehicles makes this algorithm less suited for the most applications in which multiple vehicle units are desired.

On the opposite range of the interdependence spectrum is the multiagent POMDP (MPOMDP) with a much wider coupling between agents. Indeed the modeling of the MPOMDP considers that each agent has direct access to the actions and observations of all of the other vehicles in the formation. Then, it uses the information data from the whole group to build its own policy tree. This approach makes the system not only highly dependent on the communication structure but also gives it a strong coupling in which the action of one agent has direct influence on the others. Surely this strategy has a nice fit for applications that require such strong cooperation and is able to provide perfect communication systems. As mentioned before, this is not the case for a group of multiple UAVs.

The already mentioned Dec-POMDP is located as a method that provides high dependence

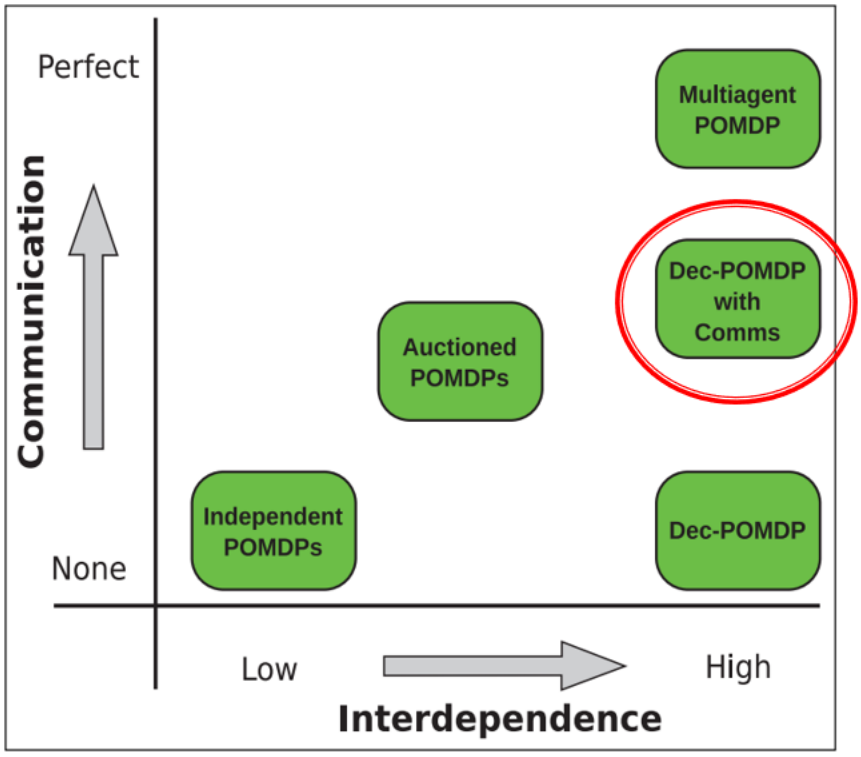


Figure 2.4: Multi-agent POMDP variations (Modified from [3])

between the agents but manages to do so without any communication whatsoever. This is possible due to the difference between the planning phase and the execution phase, i.e. during the policy optimization (performed offline) the probabilities are calculated based on the actions and observations of all the vehicles. During the execution phase, however, each vehicle has its policy already calculated and will act based only on its own observations. Although this strategy is quite useful due to the lack of communication, it still loses the aid provided by the group knowledge, even if noisy and uncertain.

To eliminate this absence while still making use of the Dec-POMDP benefits, there is the Dec-POMDP with communication method (marked with the red circle in Fig. 2.4). This procedure consists in modeling a regular Dec-POMDP with the information received from the other agents, as observations. This way, the set Z_i of possible individual observations that a vehicle might perceive, also includes the data sent by its counterparts. By applying this method, the policy building consider both the information of its own agent as well as the possible communication from the fleet. Then, the reliability on the communication is not as high as in the MPODMP case, while still getting the benefit of the group knowledge, which would not be possible with the simpler Dec-POMDP. Due to this balanced approach, the Dec-POMDP was chosen to be implemented in this work.

Finally, to complete the diagram shown in Fig. 2.4, the Auctioned POMDPs is the method developed in [9], in which during the execution phase, an auction algorithm will decide the policy of each individual agent. The policy, in its turn, has been computed in an offline planning phase prior to the execution. This way, a balanced scenario in terms of communication and interdependence can be accomplished. This method was not used in this work once it was desired, for the formation control of UAV units, that the coupling between agents was stronger than the auctioned POMDPs would provide.

2.6 UAV applications with POMDP and its extensions

Recent works have came up with novel methods to deal with UAV controls using decision making planners such as the ones based in Markov chains [20, 21, 22, 23]. Indeed those approaches are presenting to be fine solutions for complex systems that have a scholastically changing dynamics. The Partially Observable Markov Decision Process, or POMDP, is one of those techniques that accounts for both noisy observations and uncertain transition between states. As the flying scenario is usually noisy and filled with uncertainties this method suits such applications. However, it should be noticed that those work still face the lack of multi-agent control and, specifically the decentralized planning.

The Decentralized Partially Observable Markov Decision Process (Dec-POMDP) is a multi-agent decentralized extension of the single-unit planning method POMDP [18, 2]. The Dec-POMDP allows for the construction of a decision tree that dictates the individual actions of an agent based on its observations of the world states in a decentralized way, i.e. without the need of a central unit, in order to achieve the maximum global reward for the whole group [24].

The Dec-POMDP planning method can be used to construct sets of states, actions and observations that are related through stochastic matrices and, therefore, can built a system that considers uncertain measurements of the states as well as undetermined transition between them. For those reasons this framework is particularly suited for systems that should account for noisy information and require a decentralized policy, such as the UAV formation flight.

However, those methods are not decentralized, which makes the system depend on a central node for decision-making, and therefore, is very dependable on the communications and doesn't have redundancy. Thus, the Dec-POMDP might give some advances for those approaches once it gives for each individual his own policy tree for decision making.

The most notable work with Dec-POMDP applied for UAVs was done by Ragi et al. [25, 26]. Also designed for target tracking, this framework was built based on the dynamics and communication of the aircraft, which gave the overall fleet a consistent set of states, actions and observations. The problem that comes with these approaches is the fact that a high number of states is necessary and therefore might spoil the computational solving time.

To verify that problem, our previous works in [19] have came up with a system that uses the Dec-POMDP policy in an outer loop of the control system, in order to use higher level actions and, therefore, reduce the amount of states and computational time. With such system it can be integrated inner loop controllers, e.g. PIDs, that tracks the desired references, therefore taking advantage of their characteristics.

2.7 Hybrid Control

Many control approaches are being developed to use POMDP and its extensions by creating hybrid controllers, i.e. a system that uses both a continuous controller, such as a PID, and a discrete planner, such as the POMDP policy [27, 10, 28, 29], allowing to obtain the advantages that comes with both methods. The development of such hybrid algorithms can adopt the discrete policy to choose the most appropriate continuous controller (as in [27]) or to take actions in parallel to the other subsystem (such as in [10]).

The work done in [27], e.g. models a multi-layer system to control a biped platform. With suited inner-loop controllers to reliably walk on distinct ground levels, the work of Sreenath et al. uses a POMDP policy in a higher-level position to plan and choose the right controller to be implemented based on the partially observable terrain.

The work of et al. [10], however, deals with the hybrid control in a different way. The idea of this work is to model a system in which two types of actions are implemented in parallel: a discrete and a continuous. The two examples given and validated include a self-driven car, which has the discrete gears to choose from and the continuous velocities. The other example is the box push problem, in which a robot must take a box from one place to another in a specific trajectory and to accomplish that it should choose both the side of the box it must push (discrete action) and the velocity and direction of such push (continuous).

Those hybrid control systems, however, haven't been used for UAVs in an application such as the formation flight problem. More over, the approaches described are not for multi-agent problems, and much less decentralized.

Hence, there are some gaps in the decentralized formation flight that should be considered in a realistic flying scenario. The noisy and uncertain environments are crucial points for this application that can be efficiently dealt with Markov planners such as the POMDP. The decentralization is a problem solved by its extension, Dec-POMDP. However, computational time and the lack of continuous controllers are not considered using this approach alone. Therefore, an outer loop hybrid system might take all of the advantages of the discrete Dec-POMDP planner and the continuous PID controllers, with a reasonable computational time, method which has not been done for the formation flight problem of multiple UAVs.

Chapter 3

Development

3.1 Introduction

In order to achieve a formation flight of multiple fixed-wing UAVs, a hybrid control system was development. It takes the states of all the UAVs through the communication systems and decides whether to activate a discrete policy given by a decision-making algorithm such as the Dec-POMDP, or to activate the continuous PID controller in order to track a given reference and formation. Before constructing this system, a simpler method, with only two possible actions were developed to test the usage possibilities of the Dec-POMDP policy.

3.2 Problem Statement

The description of the formation flight problem and its objectives are given as follows. There are N fixed-wings UAVs flying in the three-dimensional space, with its coordinate frame being expressed in the north-east-down (NED) reference. Each agent is identified by the integer index i that ranges from 1 to N. The position of each vehicle, $x_i \in \mathbb{R}^3$ is obtained by its own built-in GPS-based localization system and it is corrupted by a zero mean white gaussian noise, $n(t)$, with variance σ^2. Therefore, $\hat{x}_i = x_i + n(t)$, in which $\hat{x}_i$ is the position seen from the agent. It also knows its own heading angle ψ_i, in respect to the North axis in the North-East plane.

The global reference is guided by a virtual leader, known to all agents, with its position represented by $x_r(t) \in \mathbb{R}^3$ and its heading angle (in respect to the North axis), ψ_r. Its trajectory is constructed with the same dynamics as the agents and has as inputs its linear and angular velocities, $v_r(t)$ and $\omega_r(t)$, respectively, as well as its altitude $d_r(t)$.

The fleet's desired formation is defined based on the difference between each vehicle's target position, $\bar{x}_i$, and the reference, i.e. $f_i(t) = \bar{x}_i - x_r(t)$. In this work, it will be considered that the magnitude of this difference will be fixed and equal to $||f_i(t)|| = \varepsilon_r$ and that the target formation is at the same altitude as the reference. Therefore the distinction between agents lies on the angular position within the perimeter of the circle of radius ε_r and the reference at its center. For this

work, an alternative formation, f_i' was also defined in case a different shape is desired during flight. This alternative parameter has the same magnitude and altitude as the main formation for all vehicles, differing therefore only by their angular positions.

Besides ε_r, it will also be defined the constants ε_d, ε_p and ε_c which are, respectively, the formation distance margin, the minimum safe distance between vehicles and the maximum communication range. If two agents are within the communication range, than they exchange their position data. By default $\varepsilon_d \leq \varepsilon_p \leq \varepsilon_r \leq \varepsilon_c$. Figure 3.1 shows a representation of those values as sets of concentric circles with the reference at its origin.

The objectives to be achieved are to take each agent i to its respective desired formation point $\bar{x}_i$, with noise and communication failure robustness, collision avoidance and little oscillation.

3.3 Dec-POMDP

Before building the actual hybrid control system to solve the problem given in section 3.2, a simpler approach was developed in order to test the Dec-POMDP planning capabilities in the multi-UAV scenario. This method was published in [19] and can be visualized in Fig. 3.2.

The problem to be solved is similar to what was previously presented but with a few simplifications such as: the space is considered to be two-dimensional so the altitude of all the vehicles are considered to be constant and identical, there is no noise in the localization system and there is no necessity that $||f_i|| = \varepsilon_r$. Other than that, the problem follows what was stated before.

To model a simple planning with Dec-POMDP, four global states, $S = \{0, 1, 2, 3\}$, were created to describe the group situations of collision avoidance, formation tracked, vehicles' approximation and communication failure, respectively. The individual observation sets are analogous to the global states but at each vehicle point of view, i.e. $z_i = 0$ means the distance between the i-th

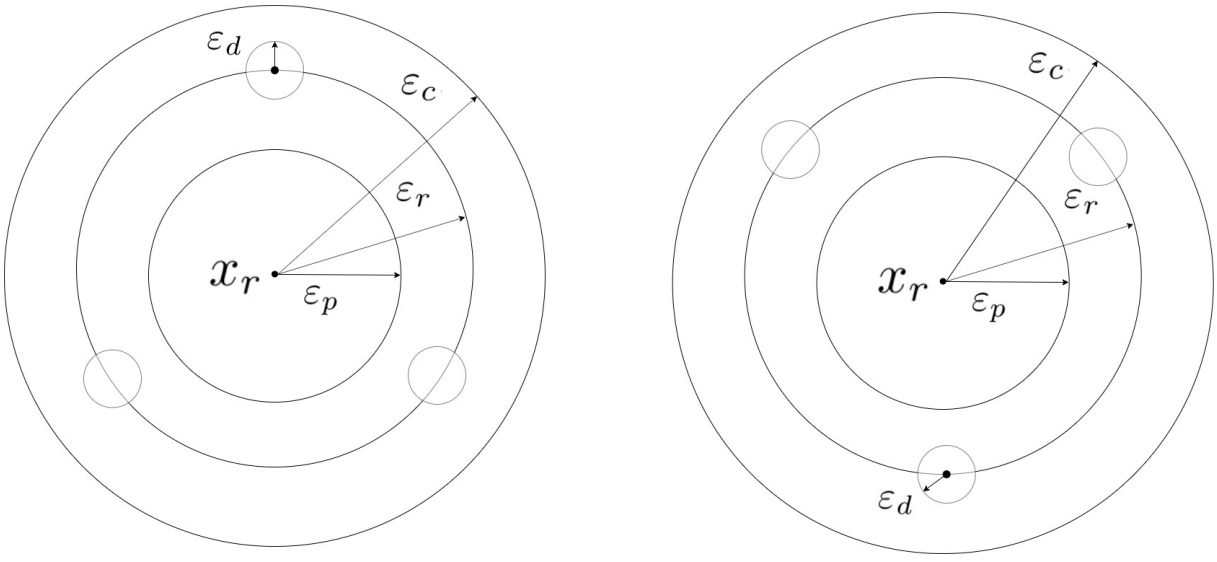


Figure 3.1: Formation limits for Dec-POMDP states

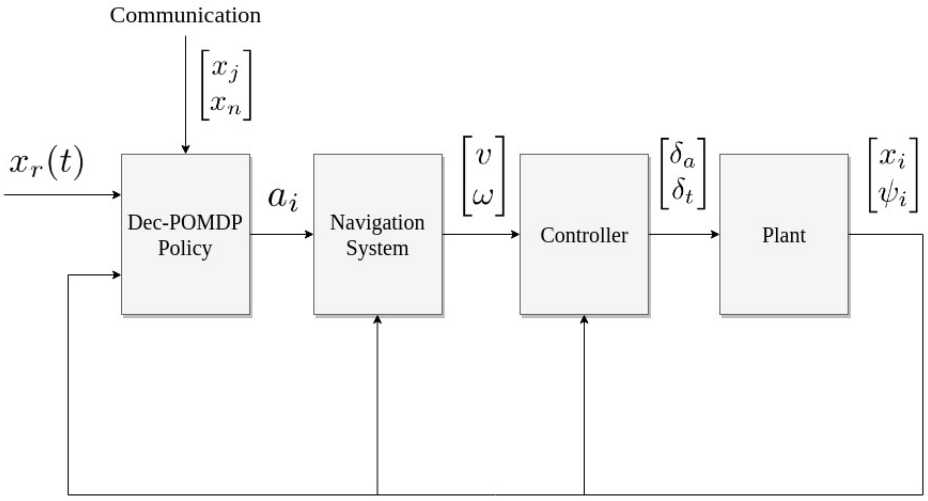


Figure 3.2: Block diagram of a UAV control system for cooperative flight

vehicle and its nearest counterpart is less than the proximity constant; $z_i = 1$ is the observation received when the vehicle is in the formation point (within the distance margin); $z_i = 2$ is for when the agent is near the formation point but not so far as to lose communication and $z_i = 3$ is when the UAV doesn't receive any data from any other vehicle whatsoever.

The set of individual actions was modeled with two possibilities: $a_i = 0$ is the action of changing the trajectory to achieve the nearest point in the formation. On the other hand, $a_i = 1$ is the action of diverging from the closest agent. A navigation system, to be further detailed, was designed in order to interpret what any of those actions means in terms of velocity changes.

The reward function, $R(s, a)$, of the system must be designed in order to represent which state should be achieved and which one must be avoided, once the states set gives the overall behavior of the group. For that reason, it was modeled only as a function of the states, $R(s)$. To appropriately weight each state, given the desired behavior of the system, this function was chosen as

$$R(s) = \begin{cases} -20, & \text{if } s = 0, \\ 50, & \text{if } s = 1, \\ 5, & \text{if } s = 2, \\ -10, & \text{if } s = 3, \end{cases} \tag{3.1}$$

as states 0 and 3 are undesired, state 1 is the system goal and state 2 is neutral.

The transition function, $T(s, a, s')$, should be designed for each joint action a. For each one, a matrix containing the transition probability between states was created. In the model designed for formation flight, each one was built following heuristic probabilities based on each action. They can be visualized in the Appendix.

Finally, the observation probability function, $O(z, a, s')$, was uniformly distributed among the possible observations, following the description of the states. E.g., state $s = 0$ means at least two agents are in danger of colliding. Given $s = 0$, than the joint observation probabilities are equally

distributed among all the combinations of two or more agents seeing $z_i = 0$. It was not a function of the actions a, once they don't affect the observation probabilities.

Fig. 3.2 presents a block diagram of each aircraft system. Starting at the outer loop is the optimum policy to be found by the Dec-POMDP solver. The description of this block can be found in Algorithm 1 which displays its pseudo-code for a system with 3 agents. It takes the states of its own vehicle, the reference trajectory, $x_r(t)$, and the positions communicated by the other agents, represented by x_j and x_n. It uses those information to generate an observation, z_i, for the Dec-POMDP policy which, in its turn, computes the actions a_i to be executed. To avoid possible intersections of states, priority to the danger and lost communication states was given. The function $\pi^*(z_i)$, in line 10, is the optimum policy obtained after solving the Dec-POMDP model in the planning phase, as described in chapter 2. Therefore, it returns the optimum action for that agent, given the observation made.

In order to translate these high-level actions into numerical commands for the flight controller, a navigation system is required. It should take the individual action outputted by the Dec-POMDP policy and convert it into linear and yaw angular velocity commands (v and ω, respectively), based on the desired points to be achieved or avoided.

The navigation model used, was created for this work, and has base values v_b and ω_b, of linear and yaw angular velocities, respectively. The output of this block is binary, i.e. resulting in a positive or negative version of the base value, depending on the action. The linear velocity has a constant component v_0 to keep the movement of the aircraft, and the base value is, then, added or subtracted to it. Therefore

$$v = v_0 + g_v(a_i)v_b, \tag{3.2}$$

in which, $g_v \in \{-1, 1\}$ is a function of the Dec-POMDP action that outputs the desired signal for navigation. If $a_i = 0$ and the desired point is ahead of the aircraft then $g_v = 1$, if it is behind,

Algorithm 1 Dec-POMDP Policy Pseudo-Code

Input: (x_r, x_i, x_j, x_n)

1: **if** $(x_j \in \varnothing \ \& \ x_n \in \varnothing)$ **then**
2: $\quad z_i = 3$ {No communication received}
3: **else if** $\min_{w \in \{j,n\}} (||x_i - x_w||) \leq \varepsilon_p$ **then**
4: $\quad z_i = 0$ {Danger of collision}
5: **else if** $\min_{f_k \in F} (||x_i - (x_r + f_k)||) \leq \varepsilon_d$ **then**
6: $\quad z_i = 1$ {Formation achieved}
7: **else**
8: $\quad z_i = 2$ {Approximation}
9: **end if**
10: $a_i \leftarrow \pi^*(z_i)$
11: **return** a_i

$g_v = -1$. The opposite goes for $a_i = 1$, with respect to the nearest agent location.

Similarly, the yaw angular velocity is given by

$$\omega = g_\omega(a_i)\omega_b, \tag{3.3}$$

in which, $g_\omega \in \{-1, 1\}$ is a function of a_i, turning the vehicle for the desired side ($a_i = 0$) or away from it ($a_i = 1$).

Finally, those instructions are passed to a flight controller that creates the corresponding throttle and aileron commands (δ_t and δ_a, respectively). The plant of the aircraft receives them and the system movement is created, generating position and angular measurements (x_i and ψ_i, respectively) that close the loop. Any flight controller that suits the aircraft model can be used for this block, once this method focus on the action planner.

3.4 Hybrid Control with Dec-POMDP

The approach used in this work is based on an hybrid control, i.e. the Dec-POMDP policy, optimized in the planning phase, works in parallel with continuous PID controllers. As each one has its own characteristics, a navigation system was built in order to define which controller should act in each situation. The designed control system can be visualized as a block diagram in Fig. 3.3.

The navigation system takes the reference and the agents data (including its own) and outputs error information for the PID Controller, the individual observation, z_i for the Dec-POMDP policy and a selecting signal to decide which of them will act based on the current situation of the fleet.

The aerodynamics control of the aircraft takes the speeds (v and ω) and altitude (d) infor-

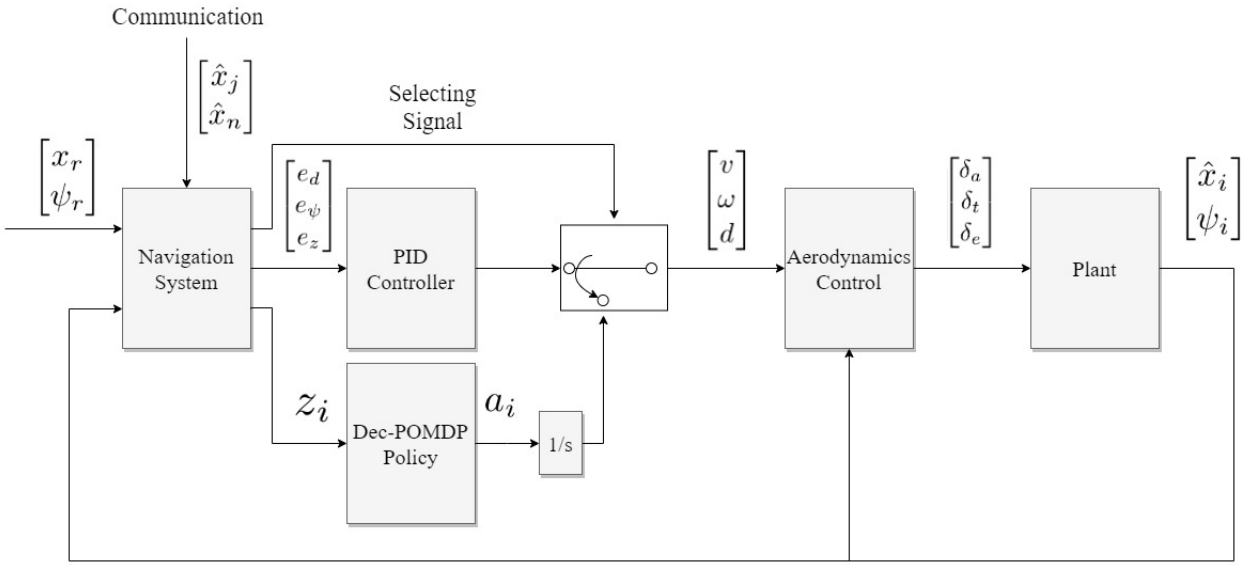


Figure 3.3: Hybrid control system diagram

mation as references and sends the corresponding responses for the ailerons, throttle and elevator commands (δ_a, δ_t and δ_e, respectively) which will then change the plane's attitude and position to be measured by the sensors.

3.4.1 Dec-POMDP modeling

To model the fleet's formation in the Dec-POMDP frame, it was considered the difference between each agent relative to the reference as well as the difference between each pair of vehicles. Figure 3.1 depicts three concentric circles with radius ε_p, ε_r and ε_c and the reference x_r at their center. Three small circles with radius ε_d in each image represent the desired formation in a group of $N = 3$ vehicles. In order to better describe the vehicles' states, it was considered one main formation (on the left) and an alternative one (on the right). This description can also be used if a different situation during flight requires an alternative geometric pattern.

State $s = 0$ was defined as the state of collision danger, in which any two pair of vehicles are distant by less or equal to ε_p. States $s = 1$ and $s = 2$ represent the formations achieved inside the circle with radius ε_r, being the first for when the agents are closer to the main desired formation and the second for the alternative one, as represented by Fig. 3.1. States $s = 3$ and $s = 4$ are achieved when all the vehicles are in their main or alternative desired positions, respectively, with a distance margin of ε_d. States $s = 5$ and $s = 6$ are analogous to states 1 and 2 but for the region between the circle of radius ε_r and ε_c. Finally, state $s = 7$ is for communication failure, i.e., when any agent loses its communication with all other vehicles.

The individual action for the agents in the Dec-POMDP policy was designed as a vector $a_i = \begin{bmatrix}\alpha_v & \alpha_\omega & \alpha_d\end{bmatrix}^T$, in which α_v is the linear forward acceleration of the aircraft, α_ω is the angular acceleration (controlled by the aileron) and α_d is the altitude acceleration (controlled by the elevator). Each of those components has three possible values, discretized by the vector $\begin{bmatrix}-1 & 0 & 1\end{bmatrix}^T$ multiplied by a base value α_{vb}, $\alpha_{\omega b}$ or α_{db}, respectively. Therefore, there is a total of $3^3 = 27$ possible individual action combinations.

Since the actions were defined as accelerations and the main inputs for the aerodynamics control are speeds, some integrator blocks should be added after the Dec-POMDP policy. The altitude command is exceptionally different since the output should be a position, which then requires a double integration.

Each agent makes an individual observation z_i of the state s, based on its own position and the communication received. Hence, a total of 8 possible individual observations can be done.

To set the transitioning function $T(s, a, s')$, a group of probability square matrices T_a, based on each action a should created, with each line for a given state s, and each column for a specific next state s'. In order to get to a reasonable stochastic relation between the states transitions, the contribution of the individual action for such change α_v, α_ω and α_d was first defined. This way, for each agent, there were three action matrices: T_v^i, T_ω^i and T_d^i. A general individual matrix is created by

$$T^i_{a_i} = \beta_i(T^i_v \circ T^i_\omega \circ T^i_d), \tag{3.4}$$

in which $T^i_{a_i}$ is the probability matrix of transitioning from state s to s', by taking the individual action a_i. The symbol $\circ$ represents the Hadamard product, in which the elements of the matrices are multiplied element-by-element. Finally, β_i is a matrix to normalize the final matrix and make all their lines sum up to 1.

Then, in order to get the global transitioning matrix T_a, considering the actions of all agents together, another Hadamard product should be computed as

$$T_a = \beta(T^1_{a_1} \circ \cdots \circ T^N_{a_N}), \tag{3.5}$$

in which β is analogous to β_i.

To properly set each matrix T^i_v, T^i_ω and T^i_d, for each agent, the possibility of transitioning were manually analyzed by using the diagram in Fig. 3.1, considering all states both in the main formation and the alternative one. This way, all possible transition between states was analyzed based on each action in order to define which transition is more possible when given this specific command. For example, by taking the action of zero linear acceleration, the probability of maintaining the current state should be greater than from changing to a different one. So this analysis were performed for all possibilities to built all of the matrices.

Setting the observation stochastic function was done as follows: as the agents become far from each other (therefore, as the states are higher), the probability that the individual observation z_i represent the state s decreases, and the same goes for the joint observation z. For state $s = 0$, the probability of the three agents to see it is set to be equal to 0.9. For only two of the agents to get it right, the probability of 0.09 was divided between all of the possible two agents permutation. And the same for only one agent, the probability if 0.01 was equality divided.

For states s ranging from 1 to 4 those values were set to 0.8, 0.15 and 0.05, respectively. For states 5 and 6, 0.7, 0.2 and 0.1 and for state 7, 0.85, 0.1 and 0.05, respectively. The increase in the probability of all the agents to see state 7 is due to the communication loss: as the UAVs become significantly distant the possibility of losing the other agent's data increase and therefore, also the probability of seeing this lack of information.

Finally, the reward function to be used in Eq. 2.8 $R(s, a)$ was built considering only the states, i.e. $R(s)$ once the fling objectives rely only on the definition of the states. Therefore, a vector R_s for the rewards was built and set as equal to

$$R_s = \begin{bmatrix} -100 & 50 & 20 & 100 & 40 & 80 & 20 & -80 \end{bmatrix}^T. \tag{3.6}$$

3.4.2 PID Control

The PID subsystem of the hybrid controller can be represented by the transfer function

$$D(s) = K_P + K_I \frac{1}{s} + K_D \frac{N_f s}{s + N_f}, \tag{3.7}$$

in which N_f os a filter coefficient. It was designed by manually tuning the K_p, K_i and K_d parameters based on the simulated response of the aircraft to the reference. The parameters were tuned to improve the tracking capabilities of each UAV model to be used, in order to obtain a faster settling time and steady state error of zero, therefore improving the results achieved in the non-hybrid approach.

Each component of the airplane control, i.e. the linear and angular speeds, had its own PID controller. The exception lies on the altitude control since the description of the problem considers that all the fleet should follow the same altitude as the reference. Once the aerodynamics control has the own altitude as input parameter, there's no need for a PID controller.

Just as in the detached case it was considered that the aerodynamics control and the plant model were already available and suited for another.

3.4.3 Navigation System

The navigation system is the block in the controller scheme that processes the internal and external data in order to define the errors and the observations, as well as the responsible for deciding which of the two previous systems should act. It follows the procedure as detailed in the pseudo-code of Algorithm 2.

Firstly, it receives information data from the reference (x_r and ψ_r), from its own internal states (x_i and ψ_i) and from the other agents (x_j and x_n, in a three agents fleet). Then, it computes the relative distances to its own position in the formations (both main and alternative ones) and with the reference (f_d, f'_d and r_d, respectively).

Determining the Dec-POMDP observation, z_i is a series of conditional statements accordingly with the definitions presented so far. Preference is given to the states of no communication and collision avoidance, as they represent situations that need fast and prioritized responses. Except for those states, the odd observations represent formations that are nearer to the main desired formation, while the even states are for when the agent gets near the alternative formation.

After the Dec-POMDP observation is determined, the errors for the PID controllers are calculated. Figure 3.4 shows a top view representation of the flight scenario, in which the i-th vehicle tracks its desired reference $\bar{x}_i$. The arrows point to the heading direction of each agent. For the linear velocity control, the error e_d is based on the distance to the desired point, i.e. f_d and the angular component $\cos(\psi_e - \psi_i)$ to account for the direction of the forward speed of the aircraft, in which ψ_e is the angle of the distance vector (in respect to the North axis). This is done in order to avoid the agent to accelerate in the wrong direction (the maximum acceleration happens when $\psi_e = \psi_i$, i.e. when the agent is heading to the desired point). The angular error was built such that the agent heading direction ψ_i is able to track both the distance angle ψ_e, in order to point in the right direction, and the reference's heading angle ψ_r, such that the path to follow

Algorithm 2 Navigation System Pseudo-Code

Input: $(x_r, \psi_r, \hat{x}_i, \psi_i, \hat{x}_j, \hat{x}_n)$

$f_d = \hat{x}_i - (x_r + f_i)$
$f'_d = \hat{x}_i - (x_r + f'_i)$
$r_d = ||\hat{x}_i - x_r||$
if $(\hat{x}_j \in \varnothing \ \& \ \hat{x}_n \in \varnothing)$ **then**
    $z_i = 7$ {No communication received}
else if $\min_{w \in \{j,n\}} (||\hat{x}_i - \hat{x}_w||) \leq \varepsilon_p$ **then**
    $z_i = 0$ {Danger of collision}
else if $||f_d|| \leq \varepsilon_d$ **then**
    $z_i = 3$ {Main formation achieved}
else if $||f'_d|| \leq \varepsilon_d$ **then**
    $z_i = 4$ {Alternative formation achieved}
else if $(r_d > \varepsilon_r) \& (||f_d|| \leq ||f'_d||)$ **then**
    $z_i = 5$ {Approximating main formation}
else if $(r_d > \varepsilon_r) \& (||f_d|| > ||f'_d||)$ **then**
    $z_i = 6$ {Approximating alternative formation}
else if $(r_d > \varepsilon_p) \& (||f_d|| \leq ||f'_d||)$ **then**
    $z_i = 1$ {In bewteen reference and main formation}
else if $(r_d > \varepsilon_p) \& (||f_d|| > ||f'_d||)$ **then**
    $z_i = 2$ {In bewteen reference and alternative formation}
else
    $z_i = 0$ {Otherwise}
end if
$\psi_e = \tan^{-1}(f_{dy}/f_{dx})$
$e_d = ||f_d|| \cos(\psi_e - \psi_i)$
if $e_d \leq 0$ **then**
    $e_\psi = (\psi_r - \psi_i)$
else
    $e_\psi = (\psi_e - \psi_i) + (\psi_r - \psi_i)$
end if
$e_z = x_{rz}$
if $z_i = 0$ **then**
    $s_s = -1$
else
    $s_s = 1$
end if
return $s_s, z_i, e_d, e_\omega, e_z$

in maintained. To avoid each UAV to spin when the reference is behind it (when $e_d \leq 0$) an if statement withdraws the term that tracks ψ_e. Finally, the altitude error is equal to the reference altitude x_{rz} once there is no PID for altitude control, as stated before.

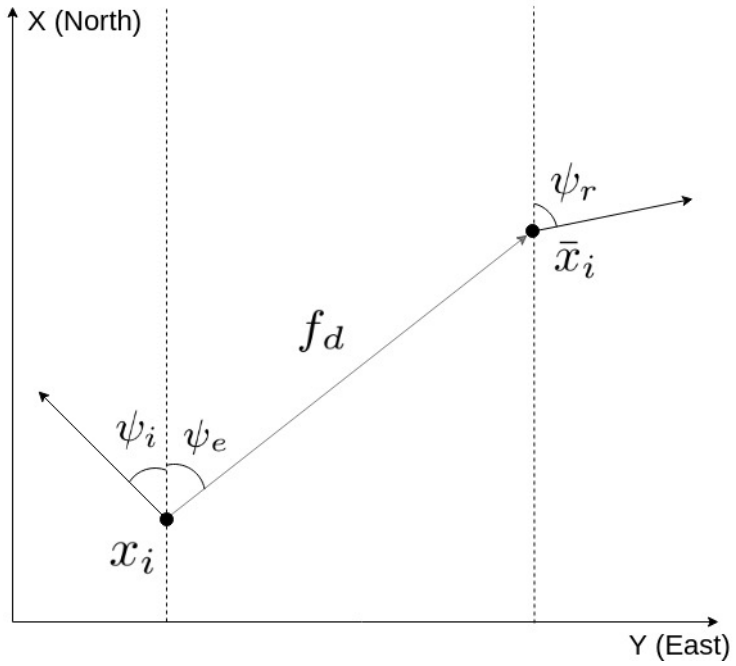


Figure 3.4: Top view representation of the vehicle tracking its reference

The selecting signal s_s is defined as equal to -1 when the Dec-POMDP policy should define the final action and equal to 1 when the PID must take control. It was concluded through the simulations that the best scenario for the hybrid control is when the Dec-POMDP policy acts in order to avoid the collision, i.e. when $z_i = 0$. Due to the fine continuous control of the PID, it is best suited for the reference and formation tracking in general.

Chapter 4

Experimental Results

This chapter presents the main simulation results achieved in both studies (Dec-POMDP and Hybrid Control). For each one, it will firstly be stated the parameters used for the Dec-POMDP policy solver and than the formation flight simulation obtained with such result will be shown. Throughout the simulations different references, formations and even UAV models were used in order to visualize the response of distinct system to the methods proposed. All the experiments were performed with $N = 3$ agents.

4.1 Dec-POMDP

The first set of results to be discussed is the one related to the single Dec-POMDP scenario, i.e. the method developed in [19] based only on the Dec-POMDP policy without a parallel continuous controller. A navigation system was also created to convert the high-level actions into actual flight commands.

4.1.1 Dec-POMDP policy solver

As mentioned in Chapter 3, the observation function was equality distributed between the possible observations that are consistent with each state. Therefore, for $s = 0$, each combination of two or more agents observing $z_i = 0$ gets a 0.1 probability (once there are 10 possible combinations for 3 agents). Similarly, given $s = 3$, only one agent is necessary to observe $z_i = 3$ and, therefore, 37 possibilities exist so each one receives a $1/37$ probability. In its turn, $s = 1$ can only be achieved if all agents observe $z_i = 1$ and, therefore, this single case gets probability 1. Analogously for $s = 2$.

The transition probability matrices were defined based on general concepts of each joint action and their results for each state transition. As $N = 3$ and there are two possible actions for each agent, a total of $2^N = 8$ matrices were built. However, the states don't differentiate one agent from the other and, therefore, some combinations of actions have the same effect on the state transition. Hence, there are only four distinct matrices which are described in the Appendix.

To solve this Dec-POMDP problem, the MADP Toolbox, developed by Oliehoek et al. [30], was used. It is an open-source library developed in C++ language to solve multi-agent planning problems such as Dec-POMDP. Several solvers are available in it but the one used in this work is JESP (Joint Equilibrium-based Search for Policies).

Due to possibly low memory issues, an agent might not be able to store a massive history of observations. For that reason, together with the low scalability in the horizon, a value of $h = 2$ was chosen for the policy search algorithm. During execution, after h time steps are passed, they start over again. The initial probability distribution, b_0, is set to be uniform for all states. The discount factor is set to $\gamma = 0.9$.

Solving the problem described resulted in the optimized value of 28.5 after 0.01 s of processing in a Intel Core i5 with 2.3 GHz and 8 GB of RAM. For comparison purposes, the method used in [26] took 5.2 s.

4.1.2 Flight Simulation

With the optimum policy π^* obtained, it can be applied in a flight model as described in Chapter 3. The experiments performed in this work were simulated in Simulink with sampling time equal to 0.01 s. The aircraft plant and controller used were the same as in [31], mainly composed of PID controllers.

As for the navigation system parameters, the standard velocity for each aircraft was set to $v_0 = 42$ m/s, the same used in [31]. On the other hand, the values used for the base velocities are $v_b = 10$ m/s and $\omega_b = 0.05$ rad/s.

The desired formation for all the experiments was

$$F = \left\{ \lambda \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \lambda \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \lambda \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right\}, \tag{4.1}$$

in which λ is a scalar factor, initially set to $\lambda = 15$. The proximity constants were set to $\varepsilon_d = \varepsilon_p = 10$ m and $\varepsilon_c = 100$ m. All of the agents were initialized at the origin and, therefore, within the communication range. Although both initial and formation conditions were set within this range, a second experiment tested communication failures as the vehicles separated from each other during simulation.

4.1.2.1 Reference Tracking

The reference, $x_r(t)$, followed similar patterns as in [7] with some scale differences to suit the aircraft model and avoid unfeasible maneuvers. In all the experiences performed, its linear velocity was set for $v_r(t) = v_0 = 42$ m/s. The first trajectory to be tested starts as a straight line pattern directed north, i.e., $\omega_r(t) = 0$. To test changes in the reference, after 25 s until termination at 100 s, its angular velocity was set to a sine wave defined by

$$\omega_r(t) = \begin{cases} 0, & \text{if } t \leq 25 \text{ s} \\ 0.05\sin(0.23t), & \text{if } t > 25 \text{ s.} \end{cases} \tag{4.2}$$

The resulting trajectories of the agents with this reference is pictured in Fig 4.1(a).

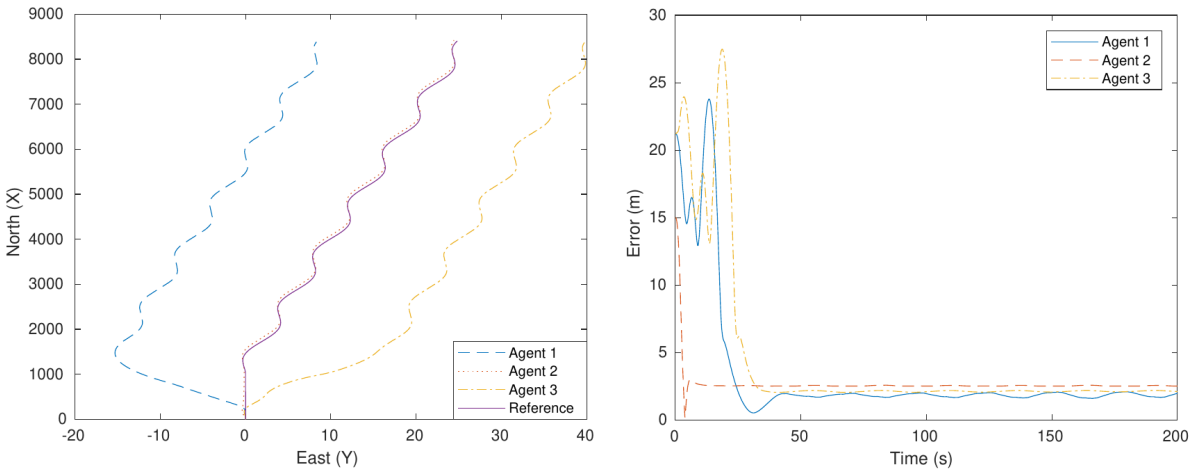
It is possible to notice that all of the agents handle the reference change, executing the same maneuver at the correct time, as required, and maintaining the pattern for the entire simulation time.

In order to verify the accuracy of the formation achieved, the errors of the agents positions with respect to their respective points in the formation can be visualized in Fig. 4.1(b). It can be seen from this result that, initially, all of them are distant from their nearest point in the formation by an amount of around 20 m. They reach the desired margin at around 25 s and stabilize the error at 50 s. Although the steady-state errors of the agents are not zero, they are within the distance margin, ε_d, established for this problem and, therefore, confirms the correct tracking of the formation and reference.

Finally, to verify the behavior of the Dec-POMDP policies, the global states, the individual observations and actions defined for the Dec-POMDP resolution can be visualized in Fig. 4.2. As expected, while the system doesn't permanently reach the desired state, $s = 1$, it oscillates between this value and $s = 0$, which represents that at least two agents are in danger of colliding, since all the agents start at the same initial position. Providentially, at around 25 s, it reaches the desired state and stays until termination.

4.1.2.2 Time-varying Formation

To test for changes in the formation, as well collision avoidance and lost in the communication, the following experiment was built: with a straight line reference for all times ($\omega_r(t) = 0$), the



(a) Trajectories

(b) Errors

Figure 4.1: Agents' trajectories and errors for sine wave reference with Dec-POMDP

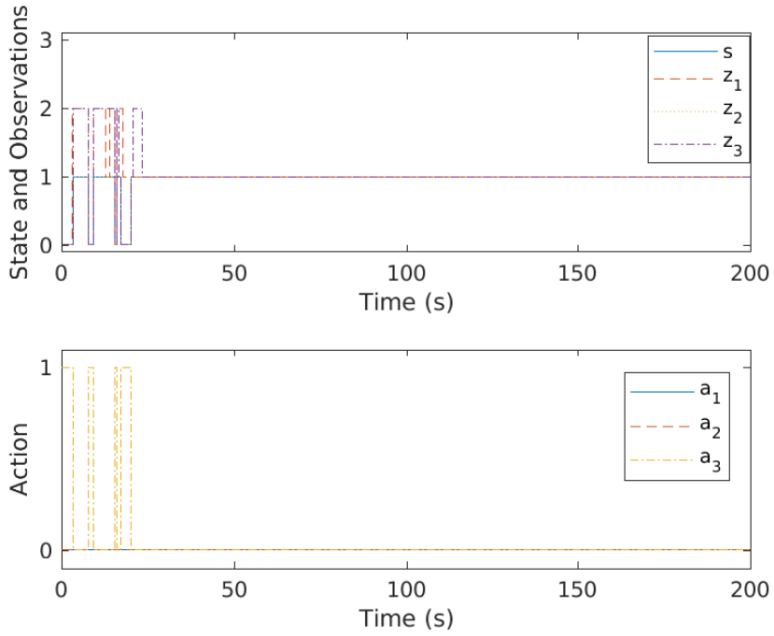
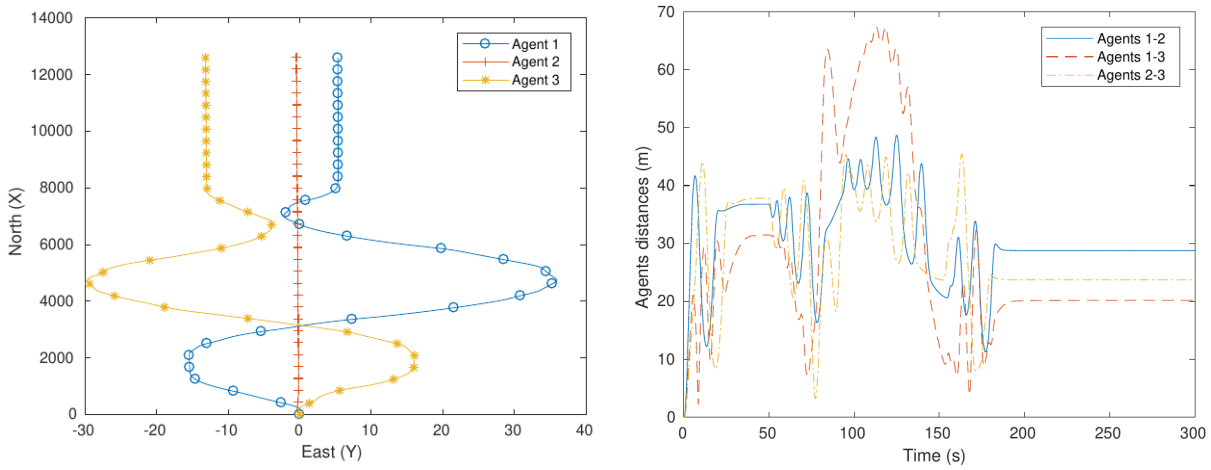


Figure 4.2: Global states, individual observations and actions for sine wave reference with Dec-POMDP

formation was initially given by Eq. 4.1 and $\lambda = 15$ for the first 50 s. After this time is passed and until termination (at 300 s) the scale parameter was changed to $\lambda = 10$. Also, the agent $i = 1$ received only the formation point $F_1 = \left\{ \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right\}$ and the agent $i = 3$, $F_3 = \left\{ \begin{bmatrix} -1 \\ -1 \end{bmatrix} \right\}$, forcing them to exchange position, passing near the vehicle $i = 2$. In order to see the system behavior with no communication, its constant was changed to $\varepsilon_c = 30$ m. The resulting trajectories can be visualized in Fig. 4.3(a). To represent the time domain of each trajectory, markings were made in Fig. 4.3(a) at every 10 s. In addition, Fig. 4.4 depicts the global states and individual observations



(a) Trajectories

(b) Distances

Figure 4.3: Agents' trajectories and distances for time-varying formation with Dec-POMDP

and actions, s, z_i and a_i respectively, reached at very time step.

The agents are able to achieve their first requested formation, just as before, reaching $s = 1$. After the 50 s are passed, agents 1 and 3 start changing position once they are no longer in their desired location and have to start the movement towards their new goal. At approximately 75 s, the agents get closer, reaching state $s = 0$ at the danger zone which makes them change their actions to avoid each other. Due to the base velocities used in the navigation system, the agents' trajectories suffered from some overshooting resulting in communication loss around 100 s. Even so, around 200 s all of them reach and stabilize their new positions within the proximity range, reaching formation once again at $s = 1$. However, it should be noted that the final formation achieved is asymmetric due to the steady state error be different than zero, as verified in the previous result.

Finally, it is possible to see that the paths of all the vehicles seem to encounter a common point. Fig. 4.3(b) displays the distances between each pair of agents as a function of time. Indeed, there are moments in which the agents are close, however, in all times they stayed at least 3 m away from each other, configuring collision avoidance for the aircraft sizes.

4.2 Hybrid Control

To test the hybrid control developed, compare it to other approaches already tested in the literature (including the previous single Dec-POMDP method) and validate the efficiency of the proposed method in different models, references and formations, three main simulations were performed. The first one is the same as [19] with a sine reference to be tracked, a follow up of this work with the controller method to improve the results achieved. The second one used the

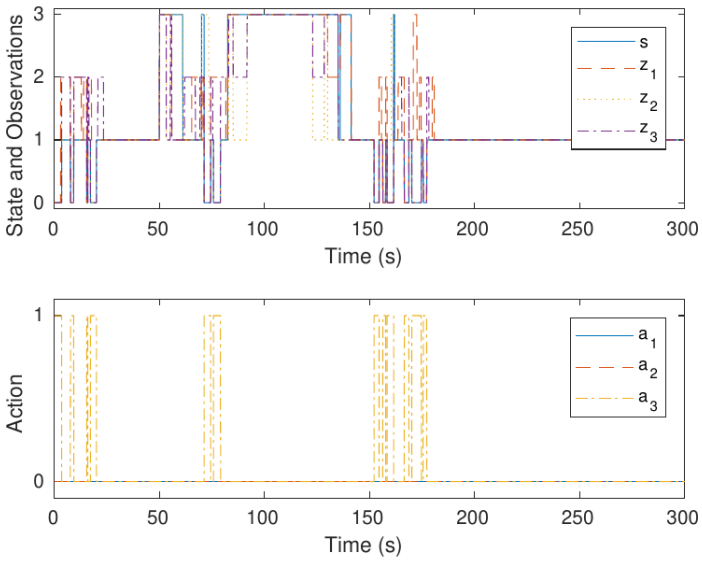


Figure 4.4: Global states, individual observations and actions for time-varying formation with Dec-POMDP

same UAV model as the previous but with a constant reference but a time-varying formation. The final simulation is based on the work done in [4]. For all of these, the exact UAV models and inner loop controllers (when present) of each one was used. Because of that difference in the models used, a PID tunning was required for each system. However, the Dec-POMDP policy was kept in both, and so as the navigation system. In both scenarios, a number of $N = 3$ UAVs were used. First it will be stated Dec-POMDP functions optimized to obtain the maximum reward policies and then the simulations will be shown.

4.2.1 Dec-POMDP Policy Optimization

To solve for the policy, the MADP Toolbox developed by Oliehoek et al. was used [30]. It is a program and library written in the C++ language in order to solve multi-agent problems related to decision-making. It solves different Markovian based planning algorithms such as the Dec-POMDP by offering distinct solving methods. The one chosen in this work was JESP (Joint Equilibrium-based Search for Policies).

Just as in [19], a horizon of $h = 2$ was chosen to avoid computational issues such as low memory or times too large to compute. The matrices used in the observations and transition functions, as well as the states rewards, were in accordance to the description given in section 3.4.1 and can be referenced in the Appendix. The initial distribution was set to be uniform for all states and the discount factor as $\lambda = 0.9$.

Solving the problem described resulted in the optimized value of 0. In all simulations the base values were set to $\alpha_v = 1$ m/s^2, $\alpha_\omega = \frac{2\pi}{100}$ rad/s^2 and $\alpha_z = 0.5$ m/s^2.

4.2.2 Sine reference

After the Dec-POMDP policy was found, it could be added to the systems desired for simulation. The first one to be tested is the system of [19]. In such work, the actions of the Dec-POMDP policy were binary (0 for collision avoidance and 1 to track the agent's reference). The navigation system, then outputs binary linear and angular velocities, taking negative or positive versions of a base value for each. It was all considered in the two-dimensional scenario.

This navigation system was replaced by the one described earlier in this work together with the PIDs controllers (without the Dec-POMDP policy, in the first moment). Once the new controller has a wilder range of values to output, it is expected that the results achieved in [19] will be improved, i.e. with better steady-state error, less oscillation and less overshooting.

With this new system built, the two main formation flights obtained in [19] were simulated in a virtual environment with a fixed-step solver using the ode4 method (Runge-Kutta) and time step equal to 0.01s. Then, as each simulation was done, the parameters K_P, K_I and K_D of the controls were individually changed to account for each contribution. This process were performed until the formation flight achieved resulted in fine reference tracking, with less error and less oscillation.

For the linear speed control, the values that better suited the model described and the control

objectives were with $K_P = 0.03$, $K_I = 0$ and $K_D = 1$. As for the angular speed PID control, the values achieved in manual tuning were $K_P = 5$, $K_I = 0$ and $K_D = 3$, with $N_f = 100$. Therefore, a PD control was already suited for such model.

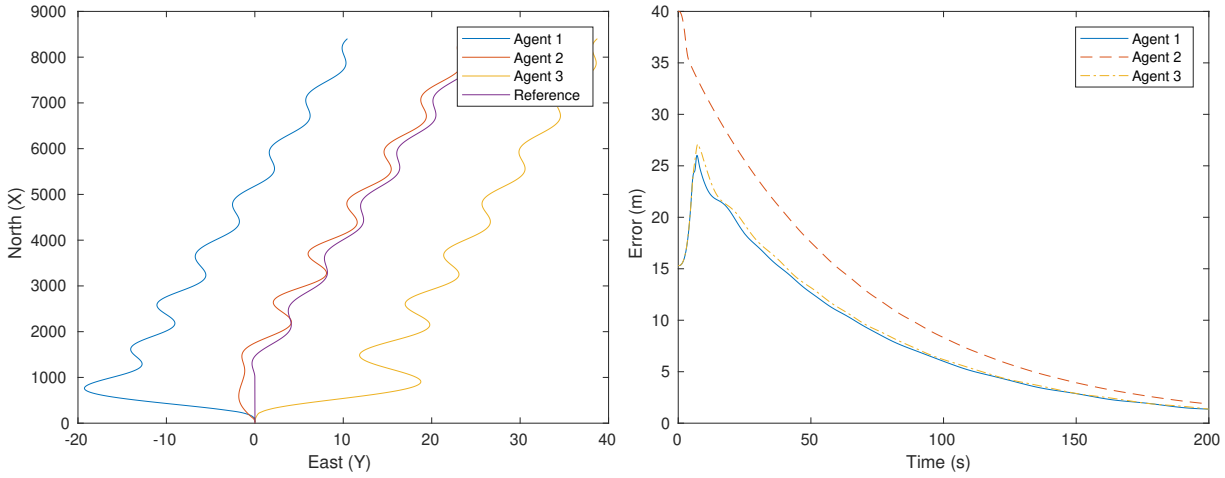
With such values, it was possible to reproduce and improve the results achieved in [19]. The first one is the sine wave reference, with constant linear velocity ($v_r = 42$ m/s) and angular speed given by $\omega_r = 0.05\sin(0.23t)$ if $t > 25$ s and 0 otherwise. The formation was kept constant and equal to $f_1 = \rho\begin{bmatrix}-1 & -1\end{bmatrix}^T$, $f_2 = \rho\begin{bmatrix}1 & 0\end{bmatrix}^T$, $f_1 = \rho\begin{bmatrix}-1 & 1\end{bmatrix}^T$ and $\rho = 15$ m at all times. The vehicles' trajectories and their respective errors are depicted in Fig. 4.5(a) and 4.5(b).

It can be noted that the agents track their formation in a stabilized way, with a declining error. The formation is kept still during flight and the vehicles leave their initial state for their desired points approaching a zero error at the steady state, showing the improved flight in comparison to the original work. However, the error decrease is significantly slower than what was achieved previously which can be explained by the PID tuning.

4.2.3 Time-varying formation

The other result that was replicated and improved with the new system, based on the previous work, is the time-varying formation, in which the formation set is equal to the previous case, except that after 50 s, agents 1 and 3 exchange their formations. The reference is kept constant ($\omega_r = 0$) with the same linear speed. The results achieved can be visualized in Fig. 4.6(a) and 4.6(b).

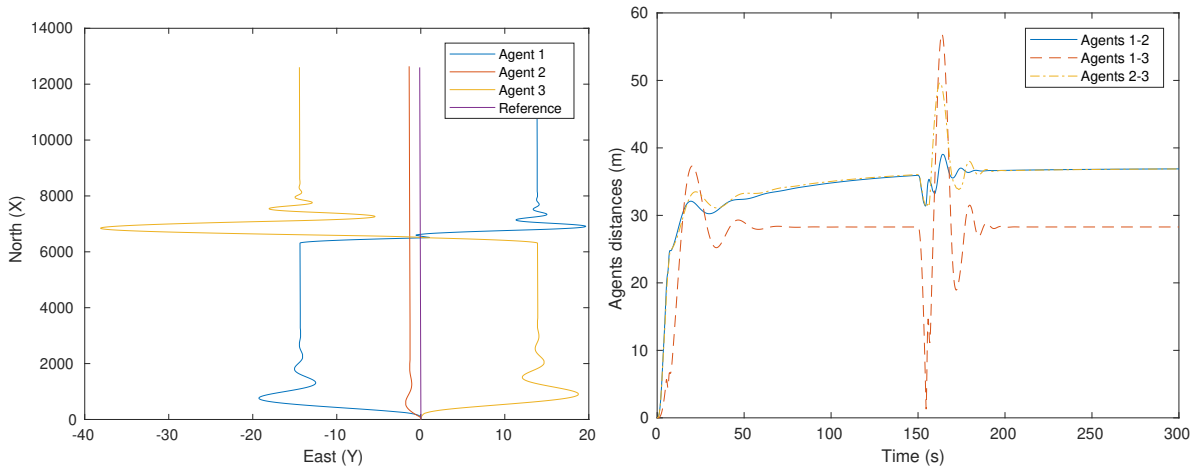
From these results, it can be noted that the formation change between agents 1 and 3 is faster than what was achieved in [19], concluding that the new system is capable of decreasing the settling time. Furthermore, after the change is concluded the formation error is significantly smaller in comparison to the previous work, leading to a more symmetric formation. In addition, the distance between the vehicles during all the task period is more stable than in the previous case while still maintaining the collision avoidance.



(a) Trajectories

(b) Errors

Figure 4.5: Agents' trajectories and errors for sine wave reference with hybrid control



(a) Trajectories

(b) Distances

Figure 4.6: Agents' trajectories and distances for time-varying formation with hybrid control

4.2.4 Step reference

To test the tracking capability of the new hybrid system, the references and formations were taken from [4] to test, as well as the UAV model and inner loop controls. Since the models were different from the ones used in the previous case, another PID tuning was required. By testing the hybrid system with the new model, the the parameter found that better track the references, with less oscillation and fast settling time were $K_P = 0.02$, $K_I = 0.00015$ and $K_D = 0$ in the linear velocity branch and $K_P = 0.2$, $K_I = 1$ and $K_D = 0$. Unlike the previous system, this model then, requires a PI control, without a derivative term.

The reference, called the ground moving target (or GMT) in such work, follows a path described by an initial heading angle of 45 degrees that suffers from a change (step) between 250 and 267 seconds of -5.7294 degrees per second. In its turn, the linear velocity starts at 2 m/s, the reference changes linearly to 10 m/s between 350 and 370 s; later on it changes again to 12 m/s between times 450 and 460 s, keeping this value until 800 s. These data can be better visualizes in Fig.4.7.

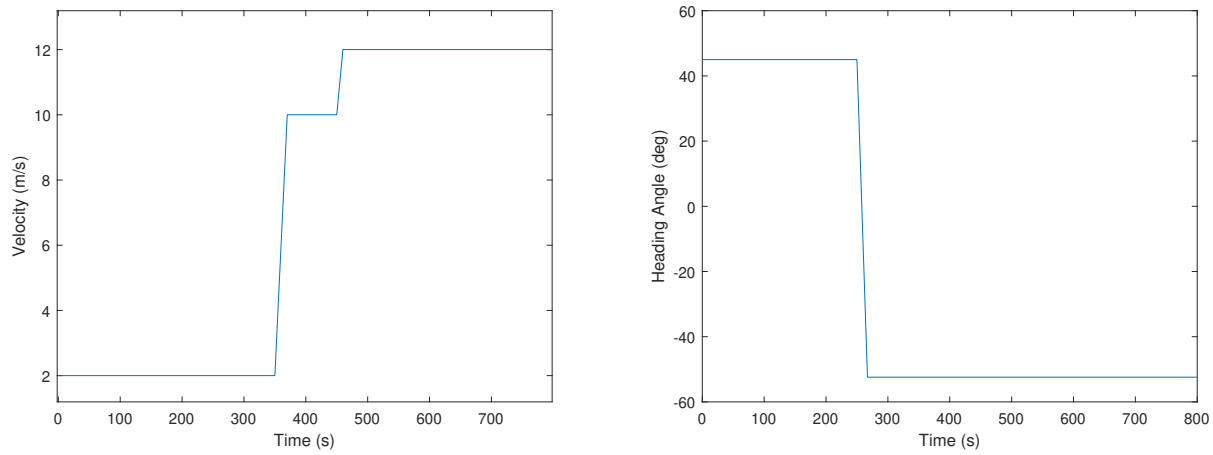
The formation desired is a equilateral triangle within a circle of radius 150 m and the reference at its center. Agent 2 is set to be right in front of the reference while agents 1 and 2 differ by 120 and -120 degrees from it within the circle, respectively.

The results achieved can be visualized in Fig. 4.8, 4.9 and 4.10. It can be seen from the results achieved that the trajectories followed by the fleet were quite similar to the path achieved in [4]. Although the error during the flight was larger than the error from [4], the trajectory was still tracked while maintaining the formation. The angle separation of each vehicle varies significantly during the flight, but reaches the desired value (120 degrees) at the end of the simulation. Another relevant point to observe is the distances between the agents to verify that, as desired, there were no collisions among them, even though the trajectories of the UAVs crosses each other at distinct times. Finally the Dec-POMDP action can be seen in Fig. 4.10, in which it can be seen that the its policy is activated around 300 s in order to avoid the collision between agents which has been

accomplished. Some differences from the original work also appeared when the vehicles followed a direct path, while in [4] they did some turn-overs. This happens only due to a navigation instruction performed in [4] to achieve a reference behind the vehicle by making a turn-over, which is performed differently for the proposed approach as explained in Chapter 3 to achieve the same goal.

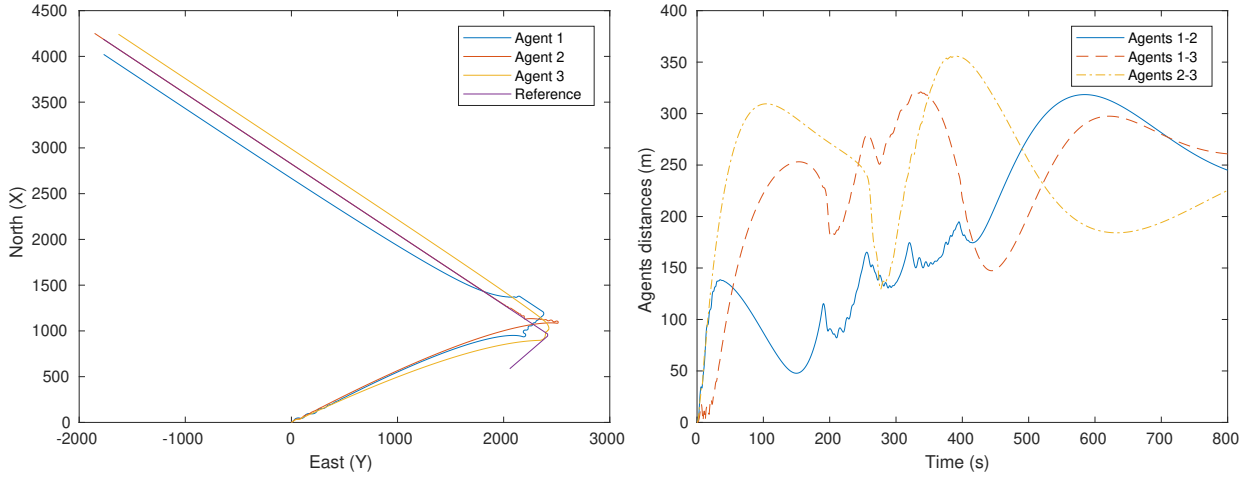
4.2.5 Noise Addition

The simulations presented so far did not considered any noise in the information received by each agent, once the work done in [4] hadn't account for it either. However, the Dec-POMDP approach, as stated in Chapter 3, was modeled to be noise robust, by considering the observation probabilities to be distributed between the states. Therefore, other simulations were performed to



(a) Velocity (b) Heading Angle

Figure 4.7: Reference velocity and heading angle



(a) Trajectories (b) Distances

Figure 4.8: Agents' trajectories and distances for reference based on [4]

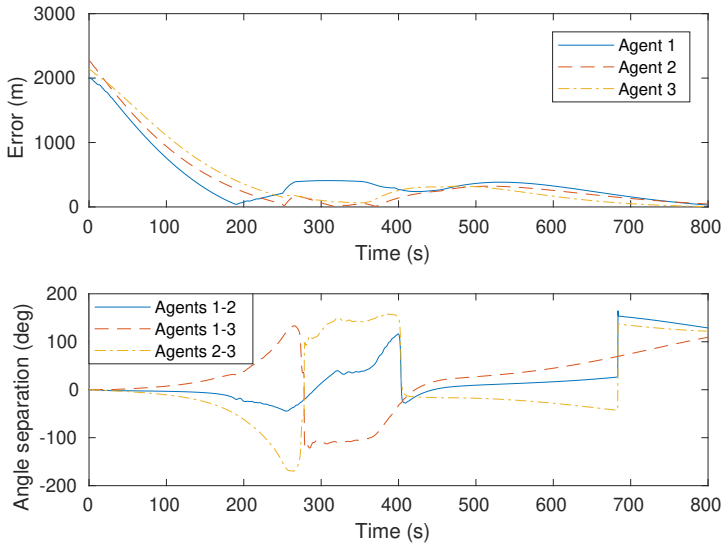


Figure 4.9: Agents' errors and angle separation for reference based on [4]

improve what was previously shown adding the common data degradation while in flight.

To model the GPS noise that degrades the vehicle's position, a band limited white noise was added to the input of each agent, not only his own but also for the data sent by the other individuals of the group. Considering that each UAV has a built-in GPS without the combination of an inertial measurement unit (IMU), then this noise can modeled with a standard deviation of $\sigma = 4$ m.

With such change, the same reference and formation used in the previous section was applied to the new system in order to compare them. The results achieved can be visualized in Fig. 4.11 and 4.12. Comparing both cases, the difference between them is quite small, with minor changes.

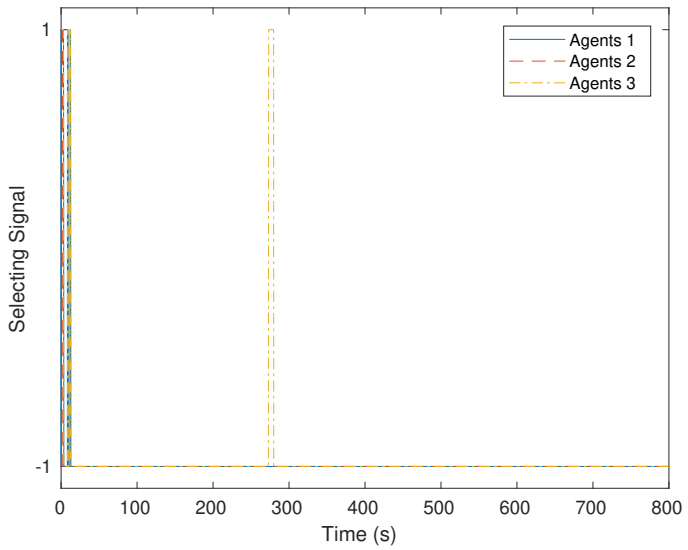
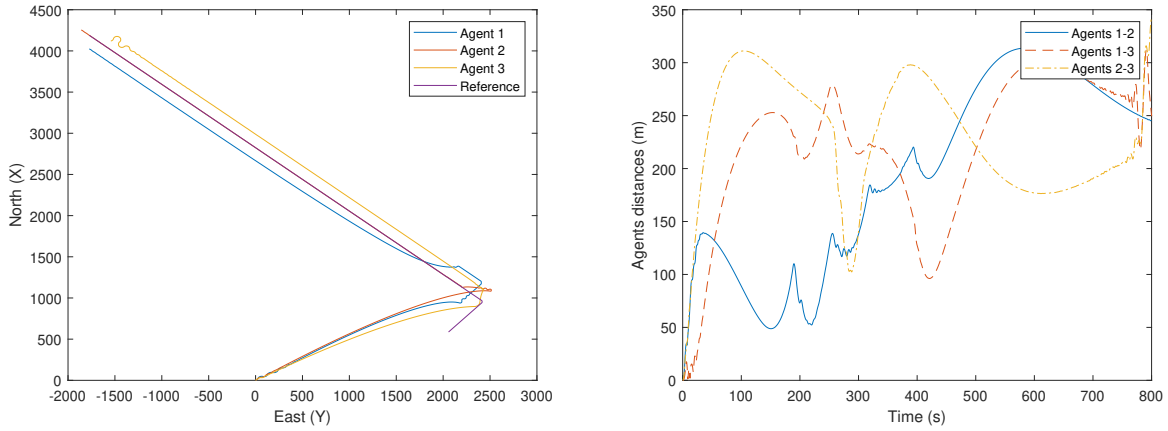


Figure 4.10: Agents' selecting signal for reference based on [4]



(a) Trajectories

(b) Distances

Figure 4.11: Agents' trajectories and distances for reference based on [4] with noise

This indicates the noise robustness that the hybrid system proposed gives to the formation flight, maintaining practically the same trajectory, distances and reference errors.

The noise robustness visualized in these results might be related to the minor changes in the observed outputs that are already capable to be corrected by the PID control. However, while dealing with the collision avoidance, the main task of the Dec-POMDP policy to correct, the wrong perception of the current state might jeopardize the mission so it is important that this algorithm is also safe to disturbed signals which is considered by the observation probability functions built in Chapter 3.

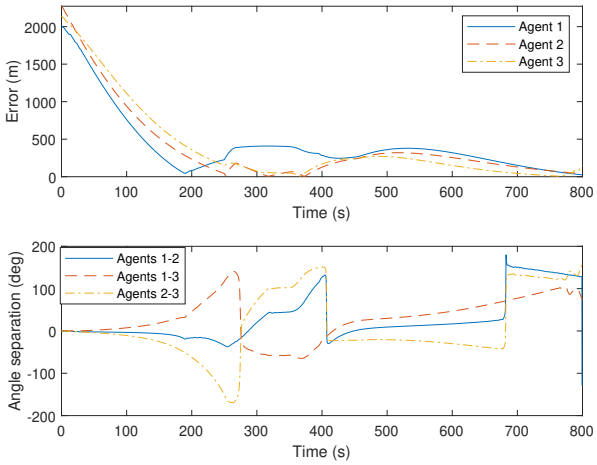


Figure 4.12: Agents' errors for reference based on [4] with noise

Chapter 5

Conclusions

The formation flight problem requires complex and dynamic objectives to be performed including reference and formation tracking, collision and obstacle avoidance, noise and communication robustness and many others. As such complex systems, the multiple UAVs scenario may require both discrete and continuous commands. Therefore, a hybrid control was proposed, in order to obtain the best benefit from both types of commands.

The Dec-POMDP method showed to be a fine planning algorithm to be used in a hybrid method. Due to its decentralized approach, this model is well suited to a flight scenario, in which the environment is very uncertain and a given UAV should not rely on central decision making unit. Indeed it was shown, through simulations that even a single Dec-POMDP policy, i.e. without a parallel continuous controller in a hybrid system, is already capable of sustaining a stable performance in flight, maintaining a fixed formation while tracking a global reference through time. The collision avoidance was also achieved with this method, which was not studied before in previous works with such algorithm. The proposed method also managed to decrease the number of states by applying the Dec-POMDP policy in an outer loop which, in its turn, improved the computational time.

However, it was possible to see that indeed the hybrid approach has several advantages in comparison to a single method, as it is able to have a finer tracking capability as well as the discrete commands in specific and emergency situations as was the case for the collision avoidance. It was shown that the hybrid strategy is comparable, and in some characteristics even superior, to other existing methods in the literature.

Although such approach is suited for all possible vehicles, in this work the formation desired was based on multiple fixed-wing aircraft. However, it is noticeable to mention that the proposed system, due to its flexibility, is able to be implemented for different vehicles such as quad-rotors, helicopters and even non-aerial vehicles and systems.

The Dec-POMDP method can be further used to many other applications within the multiple UAV scenario, due to its versatility. Through the action-state-observation sets that are constructed in the policy planning, it is possible to implement many kind of algorithms to accomplish all sort of goals. From the examples provided in Chapter 1, it can be seen that the Dec-POMDP, specially

when combined with other continuous methods, suits action planning for data acquisition, target tracking, surveillance, package delivery, traffic monitoring, footage and many other possibilities.

However, it should be noted that some disadvantages took place and can be further improved. By adopting the virtual leader method there are synchronization requirements that needs to be guaranteed, once the reference will be embedded in each vehicle, all of them should still act in harmony. Also, while using a flexible approach due to the Dec-POMDP constructions, the reference is fixed, so it can be difficult to change the route and, therefore, it might compromise a mission which requires sudden changes.

Also, in the hybrid control it can be seen that the Dec-POMDP policy only took place at very specific cases, leaving the system to be working with the PIDs most of the time. Further improvements can change those systems in order to take advantage of those policies at a longer period. In addition the collision avoidance was obtained by a narrow margin. Therefore a better maneuver from the Dec-POMDP policy can be applied at a different rate and time in order to widen this margin.

Finally, another study upon the proposed method that can be performed further is stabilization analysis. Indeed, it was not mathematically proven whether the hybrid method is actually stable or that it can guarantee collision avoidance so a detailed study should be performed.

5.1 Future work suggestions

For future works, the main follow up to what was presented in this work is the implementation of those methods in real UAVs. While the current research was allowed only for tests in simulation environments, the future developments lie on practical tests. In order to do so, the model of the real UAVs should be obtained, in order to tune the PID parameters and, consequently, avoid that the vehicles are damaged.

Besides that, suggestions for future work include the refinement of the PID controllers, possibly replacing it by another continuous system if it can provide an even better tracking capability. However, another possibility to be used is the discrete PID control. Once the system was simulated with a sampling time to mimic the continuous system, so by using all of them as discrete might also be a natural follow-up of this work. Indeed the transition matrices might also be affected by the sampling time, therefore an analysis on the impact of such period might also be necessary. Testing different actions to be performed by the Dec-POMDP policy in parallel to this other systems may also be relevant, once the multi-vehicle flight allows for many different actions to be performed.

In addition, another improvement that can be done is using an online policy, that is able to compute the decisions in real time in response to the environment. This would take away the Dec-POMDP disadvantage of separating the planning and the execution phase and, therefore allow the UAVs to plan their actions in real time. Finally, bringing together the concepts of the online planning and the increasingly researched machine learning algorithms, future works might also study the possibilities of learning algorithms within multiple UAV groups.

5.2 Acknowledgements

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

The author would like to thank the Brazilian National Council for the Improvement of Higher Education (CAPES) and the Foundation for Research Support (FAP-DF) for the support.

BIBLIOGRAPHY

[1] REN, W.; BEARD, R. W.; ATKINS, E. M. Information consensus in multivehicle cooperative control. *IEEE Control Systems Magazine*, n. April, p. 71–82, 2007.

[2] KAELBLING, L. P.; LITTMAN, M. L.; CASSANDRA, A. R. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, v. 101, n. 1-2, p. 99–134, 1998.

[3] CAPITAN, J.; MERINO, L.; OLLERO, A. Cooperative Decision-Making Under Uncertainties for Multi-Target Surveillance with Multiples UAVs. *Journal of Intelligent and Robotic Systems: Theory and Applications*, Journal of Intelligent & Robotic Systems, v. 84, n. 1-4, p. 371–386, 2016.

[4] ZHANG, M.; HUGH, H. H. Cooperative Tracking a Moving Target Using Multiple Fixed-wing UAVs. *Journal of Intelligent and Robotic Systems: Theory and Applications*, v. 81, n. 3-4, p. 505–529, 2016.

[5] CAO, Y. et al. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, v. 9, n. 1, p. 427–438, 2013.

[6] FAX, J.; MURRAY, R. Information Flow and Cooperative Control of Vehicle Formations. *IEEE Transactions on Automatic Control*, v. 49, n. 9, p. 1465–1476, 2004.

[7] MIAO, Z. et al. Distributed Estimation and Control for Leader-Following Formations of Nonholonomic Mobile Robots. *IEEE Transactions on Automation Science and Engineering*, p. 1–9, 2018.

[8] HE, L. et al. Feedback formation control of UAV swarm with multiple implicit leaders. *Aerospace Science and Technology*, Elsevier Masson SAS, v. 72, p. 327–334, 2018.

[9] CAPITAN, J. et al. Decentralized multi-robot cooperation with auctioned POMDPs. *Proceedings - IEEE International Conference on Robotics and Automation*, v. 32, n. 6, p. 3323–3328, 2012.

[10] PAJARINEN, J. et al. Hybrid control trajectory optimization under uncertainty. *IEEE International Conference on Intelligent Robots and Systems*, v. 2017-Septe, p. 5694–5701, 2017.

[11] DONG, X. et al. Time-varying formation control for unmanned aerial vehicles with switching interaction topologies. *Control Engineering Practice*, Elsevier, v. 46, p. 26–36, 2016.

[12] XUE, D. et al. Formation control of multi-agent systems with stochastic switching topology and time-varying communication delays. *IET Control Theory & Applications*, v. 7, n. 13, p. 1689–1698, 2013.

[13] DONG, X. et al. Time-Varying Formation Tracking for Second-Order Multi-Agent Systems Subjected to Switching Topologies With Application to Quadrotor Formation Flying. *IEEE Transactions on Industrial Electronics*, v. 64, n. 6, p. 5014–5024, 2017.

[14] CONSOLINI, L. et al. Leader-follower formation control of nonholonomic mobile robots with input constraints. *Automatica*, v. 44, n. 5, p. 1343–1349, 2008.

[15] YU, X.; LIU, L. Distributed Formation Control of Nonholonomic Vehicles Subject to Velocity Constraints. *IEEE Transactions on Industrial Electronics*, IEEE, v. 63, n. 2, p. 1289–1298, 2016.

[16] LEE, G.; CHWA, D. Decentralized behavior-based formation control of multiple robots considering obstacle avoidance. *Intelligent Service Robotics*, Springer Berlin Heidelberg, v. 11, n. 1, p. 1–12, 2017.

[17] SUN, Z. Cooperative Coordination and Formation Control for Multi-agent Systems. 2016.

[18] AMATO, C. et al. Decentralized control of partially observable Markov decision processes. *Proceedings of the IEEE Conference on Decision and Control*, p. 2398–2405, 2013.

[19] FLORIANO, B.; BORGES, G. A.; FERREIRA, H. Planning for Decentralized Formation Flight of UAV fleets in Uncertain Environments with Dec-POMDP. *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, p. 563–568, 2019.

[20] LIU, M.; WAN, Y.; LEWIS, F. L. Adaptive optimal decision in multi-agent random switching systems. *IEEE Control Systems Letters*, IEEE, v. 4, n. 2, p. 265–270, 2020.

[21] QIMING, Y.; JIANDONG, Z.; GUOQING, S. Modeling of UAV path planning based on IMM under POMDP framework. *Journal of Systems Engineering and Electronics*, v. 30, n. 3, p. 545–554, 2019.

[22] WANG, C. et al. Autonomous Navigation of UAVs in Large-Scale Complex Environments: A Deep Reinforcement Learning Approach. *IEEE Transactions on Vehicular Technology*, IEEE, v. 68, n. 3, p. 2124–2136, 2019.

[23] YUAN, P. et al. Markov decision process-based routing algorithm in hybrid Satellites/UAVs disruption-tolerant sensing networks. *IET Communications*, v. 13, n. 10, p. 1415–1424, 2019.

[24] AMATO, C. et al. Policy search for multi-robot coordination under uncertainty. *International Journal of Robotics Research*, v. 35, n. 14, p. 1760–1778, 2016.

[25] RAGI, S.; CHONG, E. K. UAV path planning in a dynamic environment via partially observable markov decision process. *IEEE Transactions on Aerospace and Electronic Systems*, IEEE, v. 49, n. 4, p. 2397–2412, 2013.

[26] RAGI, S.; CHONG, E. K. P. Decentralized Guidance Control of UAVs with Explicit Optimization of Communication. p. 811–822, 2014.

[27] SREENATH, K.; JR, C. R. H.; HILL, C. A Partially Observable Hybrid System Model for Bipedal. p. 137–142.

[28] LABINAZ, G.; BAYOUMI, M. M.; RUDIE, K. A survey of modeling and control of hybrid systems. *Annual Reviews in Control*, v. 21, p. 79–92, 1997.

[29] PARUCHURI, P.; BOWRING, E.; SCHURR, N. Multiagent Teamwork : Hybrid Approaches. 2006.

[30] OLIEHOEK, F. A. Tree-based Pruning for Multiagent POMDPs with Delayed Communication (Extended Abstract). *Aamas*, v. 1, n. 1, p. 1–2, 2012.

[31] CORDEIRO, T. F.; FERREIRA, H. C.; ISHIHARA, J. Y. Non linear controller and path planner algorithm for an autonomous variable shape formation flight. *2017 International Conference on Unmanned Aircraft Systems, ICUAS 2017*, p. 1493–1502, 2017.

APPENDIX

I. TRANSITION MATRICES

Here are presented the transition probability matrices used in Chapter 4.

I.1 Detached Dec-POMDP matrices

For the simpler detached Dec-POMDP approach, the transition probability matrices, $T(a)$, are described as follows. The joint action indices are decimal representations of the binary combination of individual actions (e.g. $a = 3$ means $a_3 = 0$, $a_2 = 1$, $a_1 = 1$)

$$T(0) = \begin{bmatrix} 0.6 & 0.4 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0.8 & 0.2 & 0 \\ 0 & 0 & 0.8 & 0.2 \end{bmatrix}$$

$$T(1) = T(2) = T(4) = \begin{bmatrix} 0.4 & 0.1 & 0.5 & 0 \\ 0 & 0 & 0.7 & 0.3 \\ 0 & 0 & 0.4 & 0.6 \\ 0 & 0 & 0.1 & 0.9 \end{bmatrix}$$

$$T(3) = T(5) = T(6) = \begin{bmatrix} 0.2 & 0.1 & 0.7 & 0 \\ 0 & 0 & 0.6 & 0.4 \\ 0 & 0 & 0.3 & 0.7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T(7) = \begin{bmatrix} 0.2 & 0 & 0.8 & 0 \\ 0 & 0 & 0.3 & 0.7 \\ 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

I.2 Hybrid Control Matrices

For the hybrid control system, T_v^i, T_ω^i and T_d^i, are described as follows (the number in parentheses is the index the multiplies the base values α_{vb}, $\alpha_{\omega b}$ or α_{db} of each matrix):

$$T_v^1(-1) = \begin{bmatrix} 0.2 & 0.7 & 0 & 0.05 & 0 & 0.05 & 0 & 0 \\ 0.1 & 0.2 & 0 & 0.6 & 0 & 0.1 & 0 & 0 \\ 0.6 & 0.1 & 0.2 & 0 & 0 & 0 & 0.1 & 0 \\ 0.2 & 0 & 0 & 0.1 & 0 & 0.6 & 0 & 0.1 \\ 0.1 & 0.05 & 0.6 & 0 & 0.1 & 0 & 0.1 & 0.05 \\ 0.05 & 0 & 0 & 0 & 0 & 0.2 & 0 & 0.75 \\ 0.05 & 0 & 0.05 & 0 & 0.6 & 0.05 & 0.2 & 0.05 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0.3 & 0.4 \end{bmatrix}$$

$$T_v^1(0) = 0.8I + \frac{0.2}{7}(1_{8\times 8} - I),$$

in which I is the 8-th dimensional identity matrix and 1_{8x8} is the 8-th dimensional square matrix completely filled with 1.

$$T_v^1(1) = \begin{bmatrix} 0.2 & 0 & 0.7 & 0 & 0.05 & 0 & 0.05 & 0 \\ 0.6 & 0.2 & 0.1 & 0 & 0 & 0.1 & 0 & 0 \\ 0.1 & 0 & 0.2 & 0 & 0.6 & 0 & 0.1 & 0 \\ 0.1 & 0.6 & 0.05 & 0.1 & 0 & 0.1 & 0 & 0.05 \\ 0.2 & 0 & 0 & 0 & 0.1 & 0 & 0.6 & 0.1 \\ 0.05 & 0.05 & 0 & 0.6 & 0.05 & 0.2 & 0 & 0.05 \\ 0.05 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0.75 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0.3 & 0.4 \end{bmatrix}$$

Due to the symmetry of the problem:

$$T_v^2(-1) = T_v^1(1)$$

$$T_v^2(0) = T_v^1(0)$$

$$T_v^2(1) = T_v^1(-1)$$

$$T_v^3 = T_v^1$$

$$T^1_\omega(-1) = \begin{bmatrix} 0.2 & 0.2 & 0.2 & 0.1 & 0.1 & 0.1 & 0.1 & 0 \\ 0 & 0.2 & 0.1 & 0.3 & 0 & 0.4 & 0 & 0 \\ 0 & 0.1 & 0.2 & 0 & 0.3 & 0 & 0.4 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0.4 & 0.2 & 0.3 \\ 0 & 0 & 0 & 0 & 0.1 & 0.2 & 0.4 & 0.3 \\ 0 & 0 & 0 & 0 & 0 & 0.2 & 0.1 & 0.7 \\ 0 & 0 & 0 & 0 & 0 & 0.1 & 0.2 & 0.7 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0.3 & 0.4 \end{bmatrix}$$

$$T^1_\omega(0) = T^1_v(0)$$

$$T^1_\omega(1) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.8 & 0.1 & 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0.8 & 0 & 0.1 & 0 & 0 & 0 & 0.1 & 0 \\ 0.5 & 0.3 & 0 & 0 & 0 & 0.2 & 0 & 0 \\ 0.5 & 0 & 0.3 & 0 & 0 & 0 & 0.2 & 0 \\ 0.1 & 0.5 & 0 & 0.1 & 0 & 0.2 & 0 & 0.1 \\ 0.1 & 0 & 0.5 & 0 & 0.1 & 0 & 0.2 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0.3 & 0.4 \end{bmatrix}$$

$$T^2_\omega(-1) = \begin{bmatrix} 0.1 & 0.45 & 0.45 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0.2 & 0 & 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0.2 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.2 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0 & 0.7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.3 & 0.7 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0.3 & 0.4 \end{bmatrix}$$

$$T^2_\omega(0) = T^1_\omega(0)$$

$$T^2_\omega(1) = T^2_\omega(-1)$$

$$T^3_\omega(-1) = T^1_\omega(1)$$

$$T^3_\omega(0) = T^1_\omega(0)$$

$$T^3_\omega(1) = T^1_\omega(-1)$$

$$T_d^1(-1) = \begin{bmatrix} 0.1 & 0.45 & 0.45 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0.2 & 0 & 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0.2 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.2 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0 & 0.7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.3 & 0.7 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0.3 & 0.4 \end{bmatrix}$$

$$T_d^1(0) = T_v^1(0)$$

$$T_d^1(1) = T_d^1(-1)$$

$$T_d^3 = T_d^2 = T_d^1$$