Guaranteed ellipsoidal numerical method for the stability analysis of the formation control of a group of underwater robots

Morgan Louédec

25th July 2024

Contents

Abbreviations

AUV	Autonomous Underwater Vehicle
CAPD	Computer Assisted Proof in Dynamic groups
GNSS	Global Navigation Satellite System
IMU	Inertia Measurement Units
IVP	Initial Value Problem
LMI	Linear Matrix Inequality
ODE	Ordinary Differential Equation
ROV	Remotely Operated Vehicle
USBL	Ultra-Short Baseline

Notation

t	time variable, in $\mathbb{R}$
t^+	the instant just after an update perform at time t
$\boldsymbol{x}$	continuous-time state vector, in $\mathbb{R}^n$
$\boldsymbol{f}(\ldots)$	nonlinear function of the ODE of the continuous time system
$\boldsymbol{\phi}(\ldots)$	flow of the continuous time dynamical system
$(\boldsymbol{m}_k)_{k\in\mathbb{N}}$	sequence of the discrete-time state vector, in $\mathbb{R}^m$
$\boldsymbol{h}(\ldots)$	nonlinear function of the recursive formula of the discrete-time dynamical system
$\boldsymbol{z}$	state vector of the synchronous hybrid system, in $\mathbb{R}^p$
T	period of the synchronous hybrid system
$V(\ldots)$	Lyapunov function
$[x]$	interval of $\mathbb{R}$
$[\boldsymbol{x}]$	interval vector of $\mathbb{R}^n$
$[\boldsymbol{A}]$	interval matrix of $\mathbb{R}^{n\times n}$
$[\boldsymbol{f}](\ldots)$	inclusion function
$\mathscr{E} = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma})$	ellipsoid of $\mathbb{R}^n$ with the centre point $\boldsymbol{\mu}$ and the shape matrix $\boldsymbol{\Gamma}$
$\mathcal{P}_h(\ldots)$	operator of the outer ellipsoidal enclosures algorithm with the mapping $\boldsymbol{h}$
$\boldsymbol{J} \in \mathbb{R}^{n\times n}$	Jacobian matrix

Chapter 1

Introduction

1.1 General introduction

With the development of human marine activity comes the interest in using robots to automate certain tasks. While underwater robot groups are envisaged for certain missions, these groups are not as advanced as land and air robot groups. The reason for this is that underwater constraints make it difficult to locate the robots. In this context, underwater formation control is complex and an important research subject. Various theoretical controllers have been proposed [57, 45] to make the group's behaviour more reliable.

However, new controllers need to be developed to consider the constraints that are specific to underwater robots. The control of the robots is significantly affected by hydrodynamics effects, communication issues and the heterogeneity of the group [105]. In addition, while formation controllers have been validated in simulation, there is currently a need for validation on real systems.

As for every autonomous system, one step of the validation for the formation controller is the stability analysis. A mathematical proof is required to show that the controller makes the group of robots stable, in the sense that the robots cannot go out of formation. Yet, as the complexity of these nonlinear systems increases, it becomes difficult to study the stability of a group of robots with the conventional Lyapunov method.

In parallel, the mathematical topic of Interval analysis has been shown to provide efficient numerical approaches for solving various tasks in control theory [39, 74, 10, 121]. By enclosing the solution of the computations, intervals can be used to develop numerical methods that guarantee their result. This result can so be used in a mathematical proof. Therefore, there is an interest in developing computer-assisted proofs for the study of systems that are too complex to study manually.

The main objective of this thesis is to develop guaranteed numerical methods that can assist in the proof of stability for the formation control of a group of underwater robots. This method will be designed for high-dimensional, nonlinear systems. It will be adapted to different types of mathematical models. To answer this objective, this thesis is organised as follows.

Chapter 1 presents the contributions and the context of this thesis.

Chapter 2 presents some state-of-art regarding underwater robotics and formation control. Some significant localisation and communication constraints are presented. Different types of formation controllers are classified.

Chapter 3 presents the state-of-art on the formal mathematical tools and concepts used in this thesis. It introduces the notions of dynamical systems, nonlinear stability analysis, Lyapunov theory and positive invariance.

Chapter 4 presents the numerical mathematical tools and concepts used in this thesis. It introduces the notions of interval analysis and ellipsoids. Then, some numerical methods are presented: the guaranteed propagation of ellipsoid from [95] and the guaranteed integration of the variational equation from [112].

In Chapter 5, discrete-time systems are studied. Their equilibrium point is assumed locally exponentially stable, but its domain of attraction is unknown. Based on the method from [95] a new method is introduced to compute a positive invariant ellipsoid that is part of the domain of attraction of the equilibrium point. This method is computationally tractable and can be used on high-dimensional nonlinear systems. This method first solves a discrete Lyapunov equation to find a candidate ellipsoid that is likely positive invariant. From this ellipsoid, an ellipsoidal enclosure of the reachable set is computed using the guaranteed propagation method. If this enclosure is strictly included in the initial ellipsoid, then the ellipsoid is positive invariant with respect to the system. If the inclusion is not verified, the process can be repeated by shrinking the initial ellipsoid, which reduces the pessimism caused by the non-linearity and makes the inclusion more likely to be verified. The method is then applied to a formation control example with a discrete-time model. The same example is used in the following chapters with different models.

In Chapter 6, this method is adapted for continuous-time systems. These systems are also considered locally exponentially stable. The method can compute an ellipsoid that is part of the domain of attraction. The continuous system is discretised and the method of Chapter 3 is applied to the discretised system. This is possible because the method does not require the analytical expression of the discrete mapping of the discrete system, as long as the Jacobian matrix can be computed. In the case of the discretised system, the Jacobian matrix can be computed via a guaranteed integration of the variational equation. The method is applied in the example of Chapter 5 with a continuous-time model. In addition, this chapter presents an auxiliary method that proves that the ellipsoid is positive invariant with respect to the continuous system. This auxiliary method consists in proving that the ellipsoid is positive invariant with respect to a discretisation of the continuous system with en Euler scheme.

Chapter 7 presents a generalisation of the method from [95] and those developed in previous chapters to consider singular mappings and degenerate ellipsoid. Such types of mapping can appear in a hybrid system with projection or shock effects. The method is generalised by tuning the eigenvalues of the outer ellipsoidal enclosure: the zero eigenvalues are replaced with new eigenvalues computed via interval analysis.

In Chapter 8, the methods from Chapter 5 and 6 are adapted to synchronous hybrid systems to compute a positive invariant ellipsoid that is part of the domain of attraction. The method is applied to the example of Chapter 5 and 6 with a synchronous hybrid model.

Chapter 9 presents a real case implementation of a formation control example with two underwater robots.

Finally, a concluding chapter synthesises the results presented in this thesis and describes some potential directions for future works.

1.2 Contributions

1. An guaranteed numerical method to compute an ellipsoidal domain of attraction for a high-dimensional nonlinear discrete-time system. This domain of attraction is a positive invariant ellipsoid in which the system is exponentially stable. Under review at the IEEE Transaction on Automatic Control journal. This contribution is presented in Chapter 5.

2. A variant of this numerical method to compute an ellipsoidal domain of attraction for a high-dimensional nonlinear continuous-time system. This domain of attraction is a positive invariant ellipsoid in which the system is exponentially stable. Published in [62]. This contribution is presented in Chapter 6.

3. The generalisation of the guaranteed propagation of ellipsoid from [95] to consider singular mappings and degenerate ellipsoids. Published in [63]. This contribution is presented in Chapter 7.

4. A guaranteed numerical method to compute ellipsoids in which a high dimensional nonlinear synchronous hybrid system is exponentially stable. This contribution was presented in Chapter 8.

5. Real-world experiments illustrating the robustness of formation control with two ROVs. This contribution was presented in Chapter 9.

1.2.1 Journal Publication

- Morgan Louedec, Christophe Viel, Luc Jaulin - *Computational tractable guaranteed numerical method to study the stability of n-dimensional time-independent nonlinear systems with bounded perturbation* - Automatica, July 2023, volume 153 [62]

- Morgan Louedec, Christophe Viel, Luc Jaulin - *A guaranteed numerical method to prove the exponential stability of nonlinear discrete-time systems* - IEEE Transaction on Automatic Control 2024 (under review)

1.2.2 Conference and seminary

1.2.2.1 International

- IFAC ACNDC June 2024 (London) - Morgan Louedec, Christophe Viel, Luc Jaulin - *Outer Enclosures of Nonlinear Mapping with Degenerate Ellipsoids* [63] (article)
- SWIM June 2023 (Angers) - *Propagation of degenerate ellipsoids towards computer-assisted proofs* (oral presentation)
- International Online Seminar on Interval Methods in Control Engineering - May 2023 - *Enclompassing computation of the ellipsoidal image, in the singular case* (oral presentation)
- Submeeting June 2024 (St Raphael) - *Proving the stability of a 2 ROVs formation control with ellipsoids* (oral presentation + experimental demonstration)

1.2.2.2 National

- Submeeting April 2023 (Guerledan) - *Guaranteed state prediction of a group of underwater robots, using ellipsoids and zonotopes* (oral presentation + experimental demonstration)
- Submeeting April 2022 (St Raphael) - *Guaranteed collision free trajectory tracking* (oral presentation)
- Journée GT2 Robotique Marine/Sous-marine December 2023 (Paris) - *Achieving stable formation control for two ROVs* (oral presentation)
- AID April 2023 (Paris) - *Calcul englobant de l'image ellipsoïdale dans le cas singulier* (oral presentation)
- Journée Démonstrateur angers June 2022 (Angers) - *Guaranteed collision-free trajectory tracking of a DDBOAT fleet* (oral presentation)

1.2.2.3 Laboratory

- Académie de marine February 2024
- ROBEX day 2022
- Journée pole IA & ocean 2022 and 2024

1.2.2.4 Vulgarisation

- Science en Theizh 2023 and 2024
- Fête de la science 2023

1.2.3 Awards

- Submeeting April 2022
- Young Author Award, IFAC ACNDC June 2024

1.3 Context

1.3.1 Towards a sustainable exploitation of the ocean's resources

In modern human society, the oceans abound with valuable resources, serving as a crucial reservoir for sustenance, energy, and raw materials. However, the utilisation of these resources sparks contentious discussions, particularly in the face of climate change and the UN Sustainable Development Goals 9 and 14 [81]. Ethical quandaries emerge regarding the ethical extraction of oil, gas and minerals from the seabed [106]. Anticipated expansions within the fishing and aquaculture sectors raises concerns regarding the sustainability of heightened production levels [19]. Moreover, the development of marine energy initiatives questions their environmental effect [76, 18], and the governance of these maritime assets raises issues of national sovereignty, as some countries are developing their navy [13].

Various potential future scenarios involve the creation of artificial infrastructures both offshore and along the coastlines, such as oil rigs, wind farms, aquaculture installations, underwater mining facilities, vessel constructions, as well as the laying of communication cables and pipelines. The development of these multipurpose offshore facilities also raise concerns about their viability and their ongoing maintenance requisites [2].

1.3.2 Using robots to inspect offshore structures

Human construction deteriorates over time. Offshore and underwater structures, in particular, are susceptible to fast deterioration due to the impact of pressure, salinity, and marine life [69]. Since these structures represent a high financial and material cost, it makes economic sense to extend their lifespan. To ensure the ongoing functionality of these structures, regular inspections are imperative to detect fatigue or leaks. The inspection cost would be compensated by the lifetime gain.

However, underwater inspections present greater complexity compared to those conducted on land. Traditionally, human divers have been employed for this task despite the significant risks and financial costs involved. In recent decades, there has been a growing interest in leveraging robotics to assist or even replace human divers, aiming for enhanced safety and cost-effectiveness once the technology reaches maturity [2]. Currently, several projects [11, 123] are developing autonomous inspection with a surface vehicle and an underwater robot working together, as illustrated by Figure 1.1.

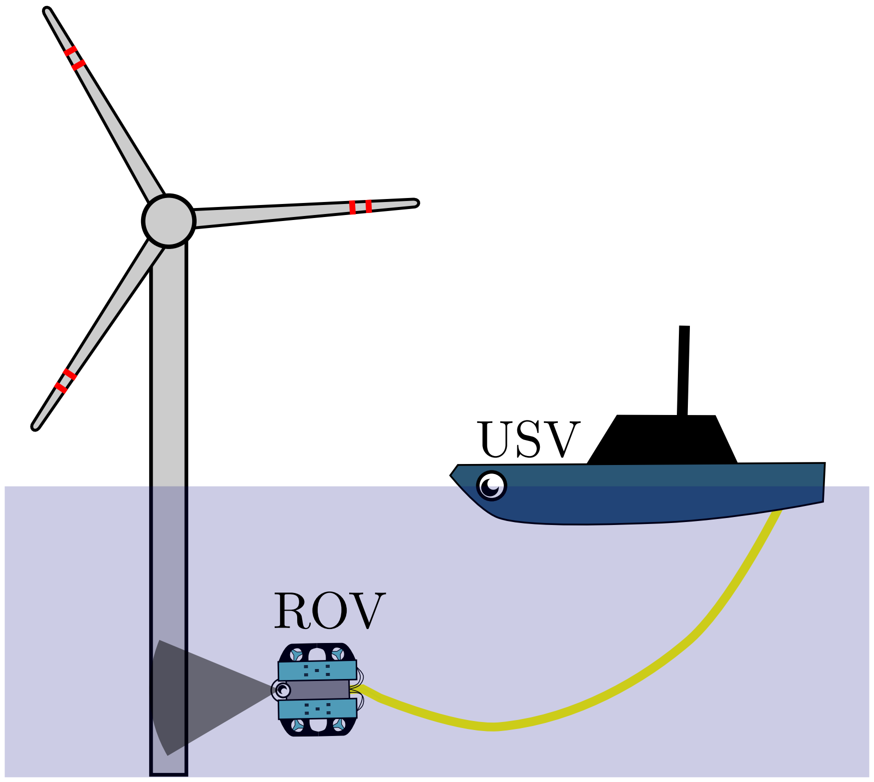


Figure 1.1: Autonomous collaborative inspection of offshore structures with an Unmanned Surface Vehicle (USV) and a Remotely Operated Vehicle (ROV)

1.3.3 The interest in using a feet of Underwater robots

As the complexity of missions grows, it becomes harder to use a single underwater robot. Employing multiple robots proves beneficial in tackling diverse challenges as presented in [118, 33, 119].

The first challenge arises with survivability, illustrated by Figure 1.2. The ocean presents a dangerous and hazardous environment for robots, subjecting them to high pressure, salt, corrosion, leaks, or collisions with rocks and unidentified objects. Consequently, there is a significant risk of losing both the robots and the valuable data they contain. When a single robot is deployed on a mission, any loss is simply unacceptable. However, with several robots on the mission, loss can be afforded. As long as the majority of the fleet survives, the data of the mission can be retrieved. Moreover, the lost robots can be saved by the rest of the fleet.

The second challenge arises with the time of the mission and the surface to cover. A fleet of robots can carry more sensors and can allocate the work to cover a larger area in less time. Moreover, the fleet can also make more complex missions relying on cooperation and flexibility. These are the benefits of teamwork.

However, while a group of robots can be more efficient, their implementation represents a technological challenge. Firstly, comparing a solitary underwater robot with a group of robots, the budget constraint means that the robots in the group are of lower quality than the solitary robot. The robots in the group are smaller and have fewer sensors, fewer tools, less battery, and less calculation capacity. Thus they are less competent than the big high-quality solitary robots. As a result, their teamwork is necessary to match the same performance as the solitary robot. Some robots from the group will also have to be specialised by holding a specific tool or sensor, making the group heterogeneous.

Furthermore, implementing the teamwork behaviours of the robots in their computer requires additional effort. The robots must know how to make a group decision when to communicate, how to avoid collision,... A fleet of robots also requires several operators and an expensive logistic, especially during launching, to prepare their mission, as the risk of failure increases with the number of vehicles.

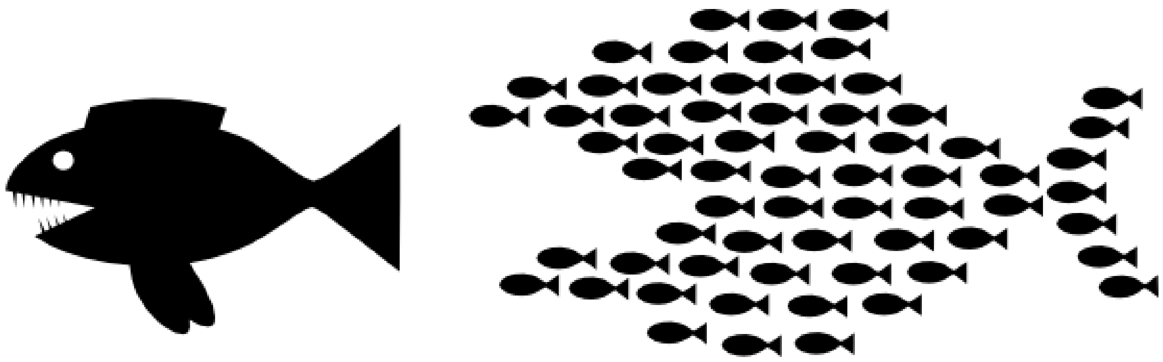


Figure 1.2: United we stand

1.3.4 The importance of stability analysis in formation control

To navigate in a group, robots must follow rules. They are expected to follow their mission, staying together and not bumping into each other or obstacles. Unfortunately, robots don't understand such an abstract demand. This is why humans must imagine mathematical constraints which will make the robots behave as a group.

In a real application of formation control, there are plenty of small physical phenomena and unpredicted external perturbations that have not been taken into account in the design of the controller, but which impact behaviour of the robots. Thus, robots may behave in a chaotic, unpredictable and dangerous way which will probably result in an outright failure of their mission, as illustrated by Figure 1.3.

To guarantee that perturbations and inaccuracies will not impact the robot's behaviour and so the maintenance of the formation, a mathematical proof is required to check the stability of each agent but also of the global system. In the presence of complex problems with a large number of robots, it is sometimes complex to prove the stability with formal methods. In this optic, this thesis provides new numerical methods to study the stability of multi-agent systems.

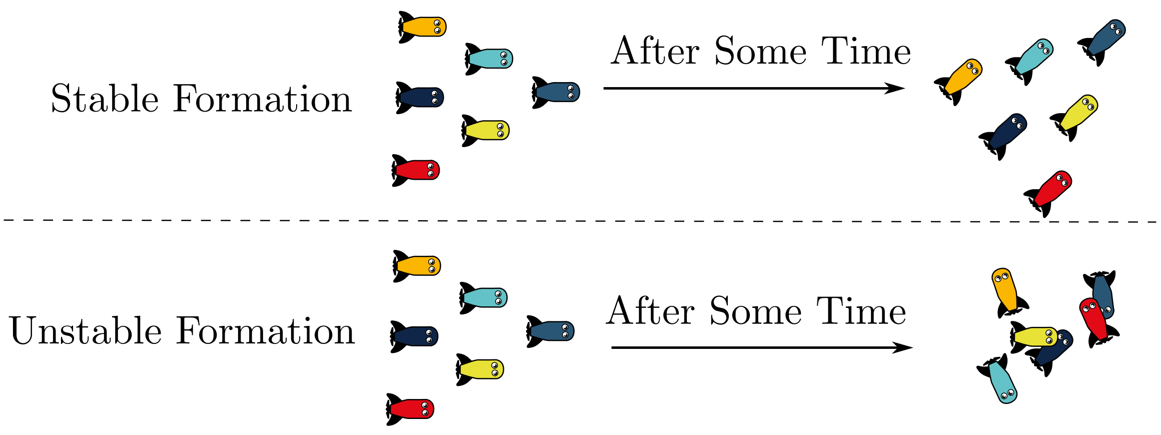


Figure 1.3: Difference between a stable formation control and an unstable one

Chapter 2

State of Art on underwater formation control

2.1 Introduction

This chapter presents the technical aspects of the systems studied in this thesis: a group of underwater robots. Section 2.2 presents the underwater robots as well as the localisation and communication constraints of the underwater environment. Section 2.3 presents several classifications of formation control that have been proposed in the literature.

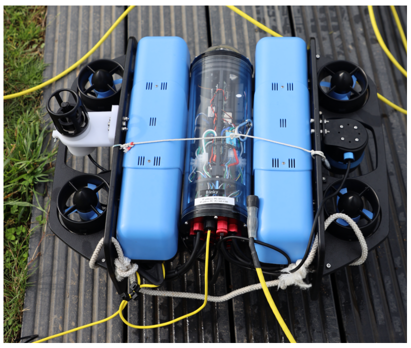
2.2 Underwater robots

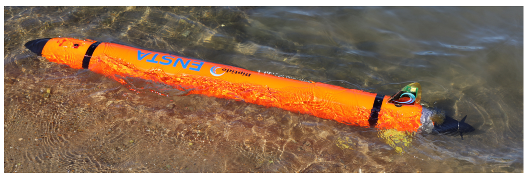
2.2.1 The realm of underwater robots

Underwater robots are usually classified in two categories:

- The Remotely Operated Vehicles (ROV) are underwater unmanned vehicles connected to the surface via an umbilical communication cable. Although they can be remotely operated by humans, some ROV are designed with some level of autonomy. Primarily used for seabed and structure inspection, ROVs are made to be highly manoeuvrable with redundant thrusters controlling the six degrees of freedom of the robot. Their modular design facilitates the easy addition of sensors, arms, and tools. Most ROV have a rectangular shape as the BlueROV2 from Figure 2.1a. Their umbilical, or tether, serves three primary functions: enable real-time bidirectional data transmission (like stream video), supply energy to the ROV, and prevent the loss of the robot during the exploration. However, the umbilical limit ROV working area has drawbacks such as collision risks, impact on ROV manoeuvrability due to umbilical inertia and drag forces, entanglement, and cable breakage.
- The Autonomous Underwater Vehicles (AUV) are underwater unmanned vehicles without umbilical cable. Since communication with a human operator

is very limited or impossible, they are designed to be fully autonomous with little communication with the surface. AUVs are also constrained by their limited battery life, which restricts mission duration. However, since their working area is not limited, they can be used for high-resolution data mapping, environmental monitoring, and infrastructure inspection. Most of the AUVs are torpedo-shaped and designed to travel long distances, as the Riptide from Figure 2.1b.


(a) The BlueROV2 from the BlueRobotics company is currently, a very popular low-cost ROV used in many Laboratories



(b) The Riptide is torpedo-shaped AUV with 3 fins and one thruster as the back

Figure 2.1: Underwater robots present at ENSTA Bretagne

In addition to the structure inspection, underwater robots are used for many more applications, as presented in [33, 119], like:

- Exploration, research and collection of all types of ocean data. They can help researchers in hydrography, oceanography, geology and archaeology. They can be used to map the seabed.
- Rescue operation and search for a missing submarine or equipment (such as the OceanGate Titan [77]), to the search of a plane wreck (such as the Malaysia Airlines Flight MH370 [101]).
- Military use as mine countermeasures, mine reconnaissance detection and localisation, surveillance, rapid environmental assessment or anti-submarine warfare.
- Mining and extraction of seabed minerals like in recent project [59].

2.2.2 Additional issues in underwater robotics

In addition to the challenge of making a group of robots, there are other common issues in the field of underwater robotics that need to be resolved if the technology is to reach maturity. These issues includes obtaining reliable and accurate robot location and robust underwater communication.

2.2.2.1 Underwater Localisation

Underwater localisation is challenging because there is no global navigational system. While mobile robotics relies primarily on a Global Navigation Satellite System (GNSS) for localisation on land, GNSS does not work underwater because seawater absorbs the electromagnetic waves used by the satellite to communicate with the robot. To overcome this problem, underwater robots can localise by fusing the the information they obtain from several sensors:

- When a robot surfaces, it can use the **GNSS** signal to know its position on Earth. Then the robot remembers its diving position. This method requires to return to the surface regularly and is therefore not applicable to deep dives, where it can take several minutes or even hours to return to the surface. Moreover, it assumes that the robot does not drift too much during the dive.
- The robot can measure precisely its depth using a **pressure sensor**. The localisation is only challenging for the horizontal position.
- When the robots are close to the seabed, they can use a **Doppler Velocity Log (DVL)** to measure their relative speed to the seabed
- When possible, the robot can also use a **Sonar** to detect or locate known underwater structures.
- The robot can estimate its movement with its **Inertial Measurement Units (IMU)**.
- Some acoustic technologies, the **Ultrashort Baseline (USBL)**, **Short Baseline (SBL)** or **Long Baseline (LBL)** can measure the position of the underwater robot at the cost of predeployed infrastructures (surface vessel or beacons)[99].

IMU This sensor, composed of a magnetometer, an accelerometer and a gyroscope, measures the attitude and the linear and angular accelerations of the robot. From this measurement, the robot can deduce its speed and position continuously. However, the localisation diverges quickly, depending on the IMU quality. A robot with a very high-quality IMU can estimate its position at about 1 cm after hundreds of kilometres travelled, but this type of IMU is larger and expensive, too much to be fitted on a fleet. So in practice, the group's robots only use its IMU to measure their attitude.

USBL Both the USBL and SBL systems consist of a transceiver and a transponder. The transceiver serves as a device capable of transmitting and receiving acoustic signals. In a USBL system, the transceiver incorporates three or more transducers, each capable of emitting signals and spaced within a baseline of 10 cm. This baseline denotes the distance between transducers. However, in SBL systems, the baseline is longer. A transponder, on the other hand, receives a signal and autonomously transmits a different one. Typically, the transceiver is situated beneath a surface vessel, while the transponder is affixed to the underwater robot. A USBL system can estimate the relative directions and distances from the surface vessel to the robot. Relative directions are determined through the phase difference of an acoustic signal across the transducer array. In contrast, in an SBL system, relative directions are estimated by measuring the time of arrival (TOA) of signals across different transducers. Ranges are then computed by measuring the time taken for the acoustic signal to travel from transmission to reception. Then, in an LBL system, three or more fixed beacons are typically positioned on the seabed. Through acoustic triangulation of the determined ranges, a robot can measure its position within the beacon area.

Selection of the sensors As explained before, the robots composing a fleet have limited space for sensors. Most of the robots have a pressure sensor and a low quality IMU because these a cheap and small sensors. Most of the AUVs have a GNSS to locate when they reach the surface. The DVL is a larger and expensive sensor that may only be equipped by the largest robots in the fleet. Sonars are also quite large and are generally equipped by robots that use them for purposes other than localisation. The USBL is about the size of a propeller and can be equipped on every robot, but can only be used when the robot does not move far away from the surface transceiver. As a result, depending on the context, underwater robots may use different sensors to localise, but localisation remains one of the major challenges for underwater robotics.

In the fleet, even though the robots have low-precision sensors, they can share their knowledge to filter information and improve the group's localisation. When the fleet is heterogeneous, the numerous small robots with low-precision sensors can use the high-precision information of the few large robots. The localisation is however often limited as described in the next section.

2.2.2.2 The wireless underwater communication

In a fleet, robots need to communicate not only to enhance their localisation but also to inform the fleet of potential dangers or to make a group decision. To communicate, cables can link mobile robots but they are subject to drag forces, to entanglement with the environment or with themselves, and they limit the robots' movements. Thus wireless communication is widely used in mobile robotics. For underwater robots, three main wireless communication technologies are proposed by the literature, as presented in [119, 32]:

- **Radio waves** are widely used in robotics for wireless communication. They have a high propagation speed, close to the speed of light. However, they are only reliable at several meters in an underwater environment due to seawater absorption. This absorption depends mainly on the conductivity of the water but also the temperature. At lower frequencies, the propagation range is increased but the data rate is reduced. A data rate between 1 to 10 Mbps can be achieved between 1 and 2 m of range.
- **Optical waves** have a higher propagation range than radio waves with 1 Gbps at 2 m making it reliable at several tens of meters. However, the turbidity

of the water and water fouling can drastically reduce the propagation range. Moreover, optical communication requires a line of sight between the emitter and the receiver, which induces a need for direction tracking to maintain it.

- **Acoustic waves** can be reliable at a far greater distance, reaching several kilometres but at the cost of a lower data rate, with 1.5 to 50 kbps at 500 m. The data rate is lower because the speed of acoustic waves is slower compared to electromagnetic waves with a 10^5 ratio. Moreover, with low speed, the Doppler effect and delays have non-negligible effects on acoustic waves. Acoustic is also subject to shadow zones, see [88]. Furthermore, low water depth can also deteriorate the signal with multi-path caused by seabed and surface echoes. In addition, acoustic transmitters can easily interfere with each other when they are in a close frequency band. In a fleet, the robots must avoid using their transmitters simultaneously.

All communication technologies are subject to noise and require specific filtering. The choice of communication technology depends on environmental constraints as well as communication requirements. Some systems may be able to use several communication methods depending on the situation, making them more reliable. In the case of an underwater fleet of autonomous vehicles, radio waves and optical waves allow a high data rate at close range and acoustic waves allow for a long communication range.

2.3 Fleet formation control

2.3.1 Classification of formation controllers

A convenient rule is to make the robots move in a predefined formation to keep them at a safe distance. In the past decade, many approaches have been considered to control the relative distances and the bearing between the robots while manoeuvring together. This is known as the formation control problem. To compare the proposed solutions, they can be classified along different criteria, which are covered by several surveys such as [118, 113, 105, 33, 119, 83, 57, 14]. First, three types of formation problems can be distinguished:

- *formation acquisition* - robots are not initially in formation and must move into formation.
- *formation maintenance* - robots start in the desired formation and must maintain it while moving as a group.
- *formation reconfiguration* - robots must change their formation shape as a reaction to task requirements such as avoiding obstacles or passing through a narrow passage.

Then, some keywords in formation control techniques can be identified:

- The *leader-follower* approach [16, 75, 66, 17, 21], illustrated by Figure 2.2. At least one robot is taken as leader and the other robots are designed as followers. The leader follows a designed reference path. The followers pursue the leader to maintain predetermined relative positions, distances or angles. This scalable approach requires a simple controller design, as the followers have the same behaviour. It is easy to implement and easy to add or remove vehicles in the fleet. However, a fault in the leader robot's behaviour leads to a chain reaction of followers.

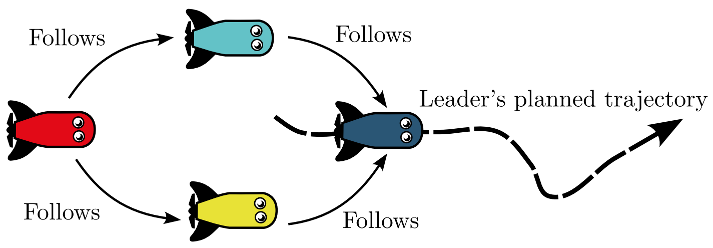


Figure 2.2: Leader-follower diamond Formation

- The *behavior-based* approach [53, 3], illustrated By Figure 2.3. Several controllers are designed for different goals such as maintaining the desired formation, avoiding collisions or avoiding obstacles. The different control inputs are summed with weight. These weights prioritise some goals over others. The resulting controller meets several control objectives simultaneously. However, the kinematic and dynamic features of the robots are not taken into consideration. As a result, the fleet's dynamic is hard to describe and the stability of the agent and the fleet is hard to guarantee.

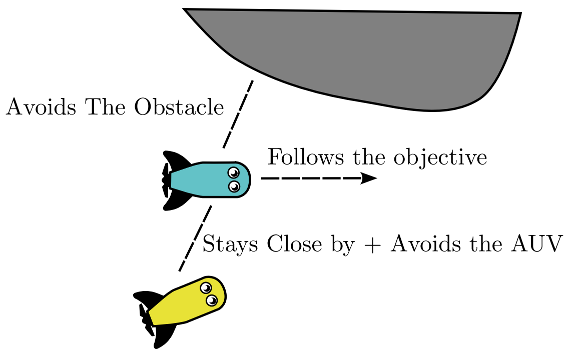


Figure 2.3: Behavior-based Formation with a sum of different common behaviours

- The *flocking* approach [96, 24], a sub-class of the behaviour-based, illustrated by Figure 2.4. Robots have simple collective behaviours with simple iterations such as the Reynolds rules. It can be seen as a sum of behaviours with artificial potential fields. It is often used to create a swarm consensus.

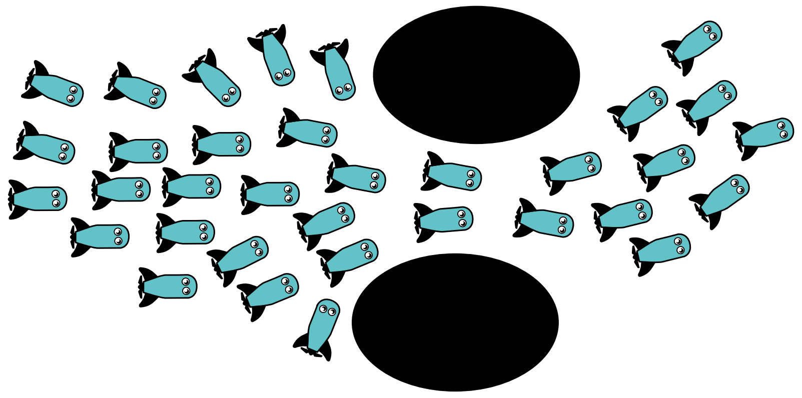


Figure 2.4: With Flocking, robots behave like a crowd

- The *virtual structure* approach [120, 55], illustrated by Figure 2.5. The fleet is considered as a single body, called a virtual structure. The virtual structure's motion and shape are planned. Then, the desired motions for the robots are determined from those of the virtual structure. It leads to better performance in maintaining formation. However, makes the formation is more inflexible which limits the ability to avoid obstacles. Moreover, if a robot is unable to follow the virtual structure, the latter doesn't wait for the robot, so the actual formation is not the desired one.

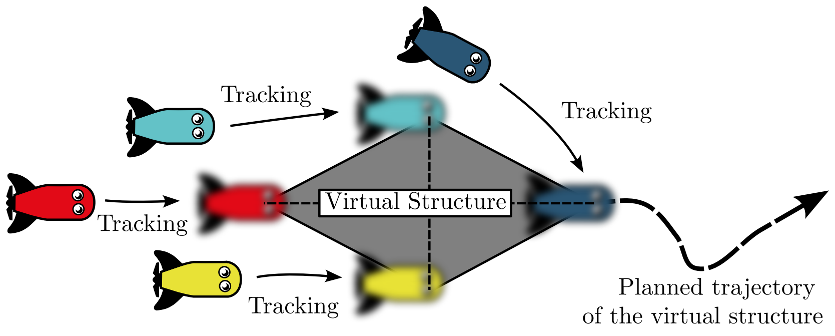


Figure 2.5: Virtual structure diamond formation

- The *artificial potential field* approach [54, 85], illustrated by Figure 2.6. Every robot follows the gradient of a prescribed potential field depending on the neighbour's position. The robots are attracted by other robots and targets, proportionally to their relative distance. They are repulsed by other robots and obstacles in inverse proportion to the distance. This approach does not need a leader robot and creates interaction between all robots, or neighbours depending on the strategy. The controllers have low computational complexity, it is simple to add or remove agents, and the fleet is more fault-tolerant. However, this approach is subject to minimum local problems resulting in unpredicted formations or unsolvable obstacle avoidance. Moreover, the obtained formation is simple and doesn't allow complex shapes.

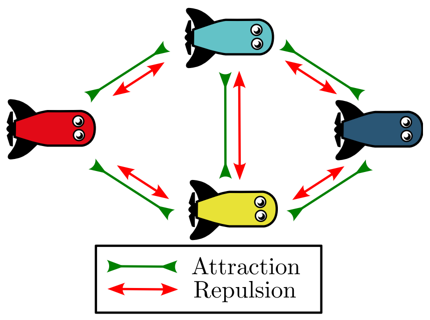


Figure 2.6: Artificial potential field diamond formation

- The *consensus* approach [84], illustrated by Figure 2.7. The robots asymptotically reach a common agreement on the formation, which can take various forms. Thanks to graph theory, it requires a simple design for linear systems. However, strong communication is required because the robots need to exchange information, unlike previous methods where formation could be maintained by observation alone.

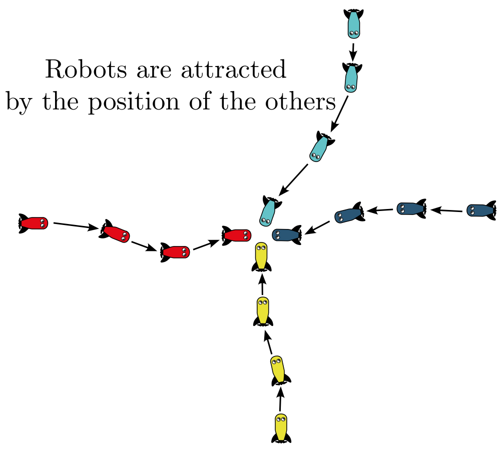


Figure 2.7: The rendezvous problem [23] is a common consensus example

The formation control approach also depends on the controlled variables. The control of the position can be classified into:

- *Position based* control [124], illustrated by the Figure 2.8. The robot's control their global positions $\left(\boldsymbol{p}_i, \boldsymbol{p}_j\right)$ with respect to a global coordinate system to track a desired position $\left(\boldsymbol{p}_i^*, \boldsymbol{p}_j^*\right)$ to achieve their desired formation. This approach is often used when the fleet navigates in a known environment to reach some referenced positions or avoid obstacles.

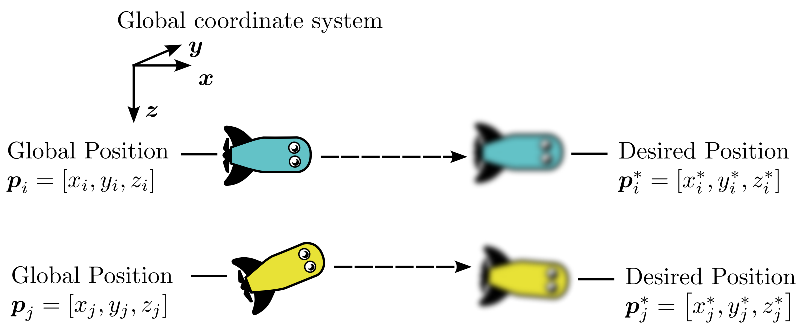


Figure 2.8: In a position-based control, robots control their global positions to track their desired position.

- *Displacement based* control [114, 104], illustrated by the Figure 2.9. Robots control their relative position $\boldsymbol{p}_{i,j}$ to some other robots to reach the desired relative position $\boldsymbol{p}^*_{i,j}$ to achieve their desired formation. This approach is often used to decouple the formation control and the navigation of the fleet. For example, in a leader-follower approach, if all robot controls their relative position with the leader, the navigation of the fleet is only controlled by the leader. This approach is also often used in unknown environments.

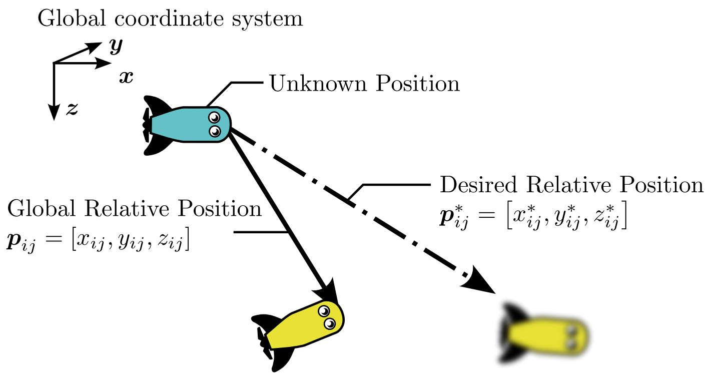

Global coordinate system
y
x
z
Unknown Position
Desired Relative Position
$p_{ij}^* = [x_{ij}^*, y_{ij}^*, z_{ij}^*]$
Global Relative Position
$p_{ij} = [x_{ij}, y_{ij}, z_{ij}]$

Figure 2.9: In a displacement-based control, robots know global orientations but only relative positions

- *Distance based* control [68, 82], illustrated by the Figure 2.10. The robots control inter-robot distances $d_{i,j}$ to achieve their desired formation. This approach is flexible in the sense that it can give rise to different formations in which the robots do not have the same place. This approach is often used in swarms to maintain the distance between the robots.

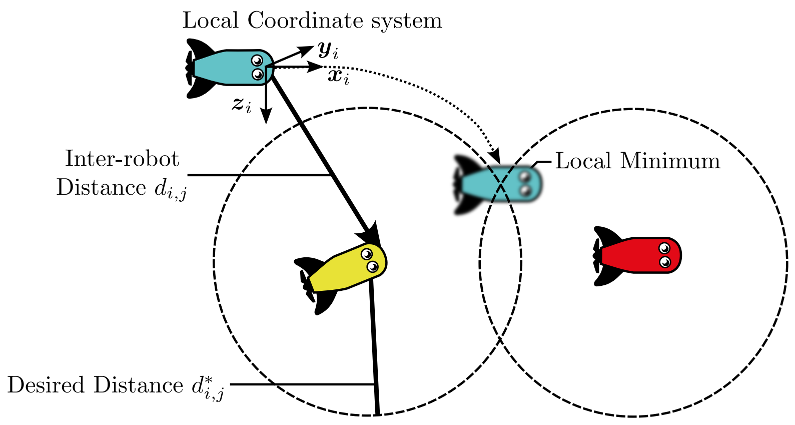


Figure 2.10: In distance-based control, robots only know local relative position and regulate their distances

The control of the attitude can also be global or relative. If a robot has 6-degrees of freedom, the position control and the attitude control can be decoupled. Note that for AUVs, the attitude often depends on the position control, as these robots have limited degrees of freedom. For example, a torpedo-shaped AUV must point toward its desired position to reach it.

The choice of the controller is also constrained by the type of information the robot can sense or communicate. Having more information requires more sensing or communication capacity. For most formations, knowing the relative pose between the robots is sufficient.

2.3.2 Communications

Finally, Formation controls can be classified based on the communication architecture:

- *Centralised* [22], illustrated by the Figure 2.11. The robots can only communicate with a master. The master can be one of the fleet's robots or an external agent (buoy, surface boat, land computer, satellite,...). The master knows all the information about the fleet and sends the objectives to the fleet. Centralised architecture is easy to implement but has weak robustness regarding faults of the master robot. The fleet size is also limited by the master's communication range and computing capacity. The master may also be saturated by too many messages.

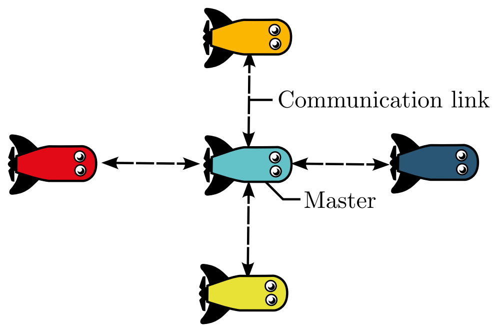


Figure 2.11: Example of centralised communication architecture

- *Hierarchical* [60], illustrated by the Figure 2.12. It is an extension of the centralised architecture with sub-controllers. These sub-controllers process the command from the centralised controller and transmit new commands to their own cluster of the fleet. Then, they also give feedback to the centralised controller. This architecture increases the scalability of the centralised controller but is still not robust regarding fault in the high hierarchy. The communication chain can also break partially when one of the masters fails.

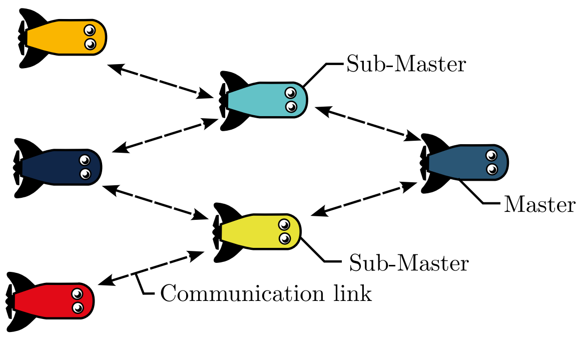


Figure 2.12: Example of hierarchical communication architecture

- *Decentralised* [53], illustrated by the Figure 2.13. Robots have no master and make their own decisions based on the local information they can perceive. This architecture overcomes the issues of centralised formation control. However, the absence of communication between the robots limits robot cooperation.

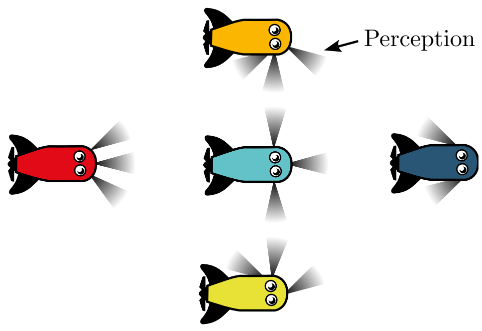


Figure 2.13: Example of decentralised communication architecture, where there is no communication between robots

- *Distributed* [56], illustrated by the Figure 2.14. robots make they own decisions but they can also communicate between them and share local information with their neighbours. The formation is maintained with each robot synchronising with its neighbours. This architecture has a better robustness and scalability than the centralised one but requires more overall communications and computing.

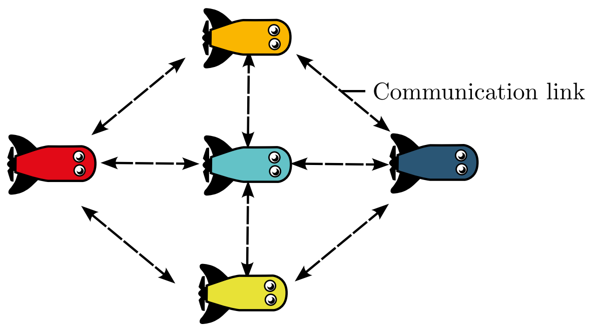


Figure 2.14: Example of distributed communication architecture

2.4 Conclusion

As presented in Section 2.3, there is a large variety of formation controllers. Unfortunately, underwater formation control is still in the early stages. Theoretical solutions are developed under some strong assumptions which are difficult to realise in underwater environments [119], making formation control ineffective in practical applications. More non-neglectable constraints should be considered such as environmental disturbance, communication loss, delays [110, 117], actuators saturation, unknown parameters in the dynamical model... Having these constraints will make theoretical solutions more reliable in practice, but it will be more complex to study robustness, stability, collision avoidance and obstacle avoidance. The next chapter will present the classical formal stability analysis methods which give solid stability proofs but are difficult to use for complex systems.

Chapter 3

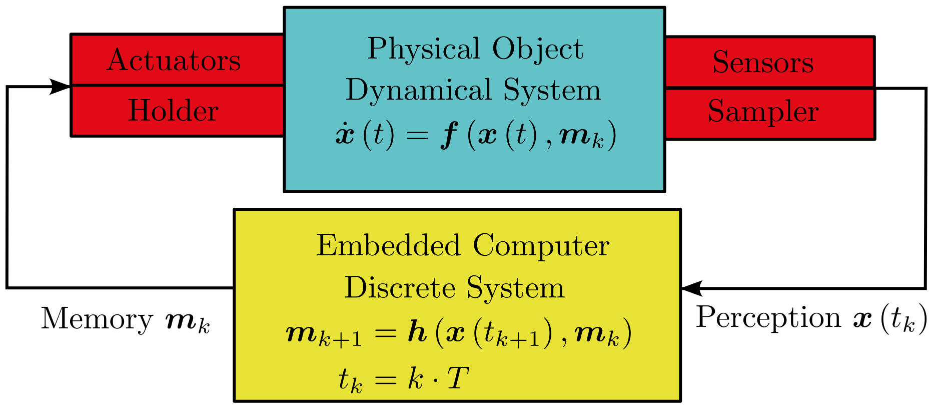
State of Art on formal stability

3.1 Introduction

This chapter presents the formal stability analysis background used in this thesis. To study the stability of a group of robots, one must first propose a mathematical model that describes the group's behaviours. These models are nonlinear dynamical systems and are presented in Section 3.2. Then, the classical methods to study the stability of these systems, based on the Lyapunov theory, are presented in Section 3.3.

3.2 Dynamical Systems

A robot is a machine capable of moving in its environment according to what it perceives. The robot's motion is decided by an on-board computer, which processes the information of the sensors and controls the actuators. In this way, the behaviour of the robots is determined both by algorithms and by the laws of physics. Consequently, a group of robots can be described as a cyber-physical system [100, 80], as in Figure 3.1. This cyber-physical system is composed of a continuous-time dynamical system presented in Section 3.2.1 and a discrete-time system presented in Section 3.2.2. In the automation field, cyber-physical systems are mathematically described as hybrid systems which are presented in Section 3.2.3.



Actuators
Holder
Physical Object
Dynamical System
$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{m}_k)$
Sensors
Sampler
Embedded Computer
Discrete System
$\boldsymbol{m}_{k+1} = \boldsymbol{h}(\boldsymbol{x}(t_{k+1}), \boldsymbol{m}_k)$
$t_k = k \cdot T$
Memory $\boldsymbol{m}_k$
Perception $\boldsymbol{x}(t_k)$

Figure 3.1: Cyber-Physical model of a group of robots. The sensors, computers and actuators form a feedback loop that is used to control the robots. The physical part of the system is described by a continuous-time nonlinear dynamical system with the state variable $\boldsymbol{x}(t)$. The computers of the robots act as a discrete-time system with the state variable $\boldsymbol{m}_k$.

3.2.1 Continuous dynamical systems

The theory on dynamical systems is taken from [37, Section 7]. The mechanical behaviour of the robots is generally described by an *Ordinary Differential Equation* (ODE) such as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t)) \tag{3.1}$$

where $t \in \mathbb{R}$ represents the time variable, $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a nonlinear function and $\boldsymbol{x}(t) \in \mathbb{R}^n$ represents the state of the robot at the time t. Since (3.1) fully describes the system's behaviour from the current state, this system of differential equations is said to be an *autonomous system*. The function $\boldsymbol{f}$ can also depend on the time t or on an input $\boldsymbol{u}(t) \in \mathbb{R}^n$, which can be used to take external commands into account. The function $\boldsymbol{f}$ can also be interpreted as a vector field that the state $\boldsymbol{x}$ follows.

Sometimes, the solutions of (3.1) may only be defined locally in time to ensure the existence and uniqueness of solutions. *Local dynamical system* are defined from *Initial Value Problem* (IVP) presented by Definition 3.1.

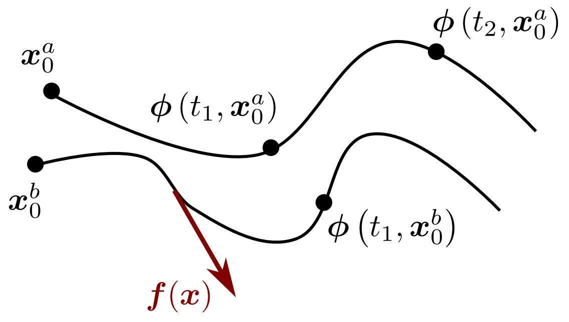
Definition 3.1. (IVP) Solving the IVP consists in finding the state trajectory $\boldsymbol{x}$ that verify the ODE (3.1) and the initial condition $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ where $t_0 \in \mathbb{R}$ and $\boldsymbol{x}_0 \in \mathbb{R}^n$.

Unfortunately, most nonlinear systems of differential equations are impossible to solve analytically. In other words, having an analytical expression for solution of (3.1) $\boldsymbol{x}(t)$ is rare.

Then, to describe the behaviour of the system for every initial condition $\boldsymbol{x}_0$, the ODE (3.1) is associated with a dynamical system. A *dynamical system* is a mathematical model that describes how a point in a given space evolves over time according to a specific rule. The definition of *continuous-time dynamical systems* is given in the following definition 3.2, illustrated by Figure *3.2*.

Definition 3.2. A continuous-time (or smooth) dynamical system is a continuously differentiable function $\boldsymbol{\phi} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ where $\boldsymbol{\phi}(t, \boldsymbol{x}_0) = \boldsymbol{\phi}_t(\boldsymbol{x}_0)$ satisfies

- $\boldsymbol{\phi}_0 : \mathbb{R}^n \to \mathbb{R}^n$ is the identity function
- $\boldsymbol{\phi}(t, \boldsymbol{\phi}(s, \boldsymbol{x}_0)) = \boldsymbol{\phi}(t + s, \boldsymbol{x}_0), \forall t, s \in \mathbb{R}, \forall \boldsymbol{x}_0 \in \mathbb{R}^n$.



$\boldsymbol{\phi}(t_2, \boldsymbol{x}_0^a)$
$\boldsymbol{x}_0^a$
$\boldsymbol{\phi}(t_1, \boldsymbol{x}_0^a)$
$\boldsymbol{x}_0^b$
$\boldsymbol{\phi}(t_1, \boldsymbol{x}_0^b)$
$\boldsymbol{f}(\boldsymbol{x})$

Figure 3.2: Illustration of a continuous-time dynamical system with $t_2 > t_1$. The trajectories passing by $\boldsymbol{x}_0^a \in \mathbb{R}^n$ and $\boldsymbol{x}_0^b \in \mathbb{R}^n$ are described by the flow function $\boldsymbol{\phi}$ and follow the vector field $\boldsymbol{f}(\boldsymbol{x})$.

For the sake of simplicity, in this thesis, continuous-time dynamical system will sometimes be referred to as a continuous system. In general, a continuous-time dynamical system is related to the ODE (3.1) with the notion of **flow** given in Definition 3.3.

Definition 3.3. (flow) A continuous-time dynamical system $\boldsymbol{\phi} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ is the **flow** of the ODE

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) \tag{3.2}$$

with $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ if it verifies

$$\forall t \in \mathbb{R}, \forall \boldsymbol{x}_0 \in \mathbb{R}^n, \frac{\partial \boldsymbol{\phi}}{\partial t}(t, \boldsymbol{x}_0) = \boldsymbol{f}(\boldsymbol{\phi}(t, \boldsymbol{x}_0)). \tag{3.3}$$

The flow $\boldsymbol{\phi}$ can describe the solutions of the ODE, such that for any $(t, \Delta t) \in \mathbb{R}^2$

$$\boldsymbol{x}(t + \Delta t) = \boldsymbol{\phi}(\Delta t, \boldsymbol{x}(t)). \tag{3.4}$$

The ODE function $\boldsymbol{f}$ can also be defined from a flow $\boldsymbol{\phi}$, as

$$\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n, \tag{3.5}$$

$$\boldsymbol{x}_0 \mapsto \frac{\partial \boldsymbol{\phi}}{\partial t}(0, \boldsymbol{x}_0). \tag{3.6}$$

When $\boldsymbol{f}$ is nonlinear, it is not always possible to find an analytical expression for $\boldsymbol{\phi}(t, \boldsymbol{x}_0)$. The following example shows an approximation of the flow when an analytical expression cannot be found.

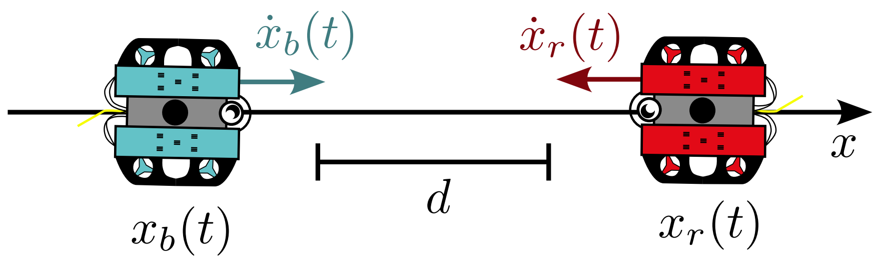


Figure 3.3: Illustration of the example. The robot must reach a distance consensus.

Example 3.1. To illustrate the continuous-time dynamical system, imagine two robots, *Blue* and *Red*, as illustrated by Figure 3.3. They can move along a line. They are controlled to maintain a desired distance $d > 0$ between them so that they can see each other. The position of *Blue* is written $x_b(t) \in \mathbb{R}$, the position of *Red* is written $x_r(t) \in \mathbb{R}$.

Considering the distance error

$$e(t) = x_r(t) - x_b(t) - d, \tag{3.7}$$

the objective of the robots is to reduce this error to zero. Assume that the speed of the two robots is controlled such that

$$\begin{cases} \dot{x}_r(t) &= -\arctan(e(t)), \\ \dot{x}_b(t) &= \arctan(e(t)). \end{cases} \tag{3.8}$$

This system can be described by the state vector

$$\boldsymbol{x}(t) = \begin{bmatrix} x_r(t) & x_b(t) \end{bmatrix}^T \in \mathbb{R}^2, \tag{3.9}$$

and the ODE

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t)), \\ &= \arctan(e(t)) \cdot \begin{bmatrix} -1 \\ 1 \end{bmatrix} \\ &= \arctan\left(\left(\begin{bmatrix} 1 & -1 \end{bmatrix} \cdot \boldsymbol{x}(t) - d\right)\right) \cdot \begin{bmatrix} -1 \\ 1 \end{bmatrix}. \end{aligned} \tag{3.10}$$

To our knowledge, there is no analytical expression for the solutions of (3.10). Thus there is no known analytical expression for the flow $\boldsymbol{\phi}$ of this dynamical system. However, given a initial condition $\boldsymbol{x}_0 \in \mathbb{R}^2$ and a time $t \in \mathbb{R}$, one can approximate $\boldsymbol{\phi}(t, \boldsymbol{x}_0)$ by an Euler Scheme. As illustrated by Figure 3.4, the trajectories of the system converge towards the line of equation $e = 0$ which is the objective of the robots. Note that in this example, the formation control is a consensus.

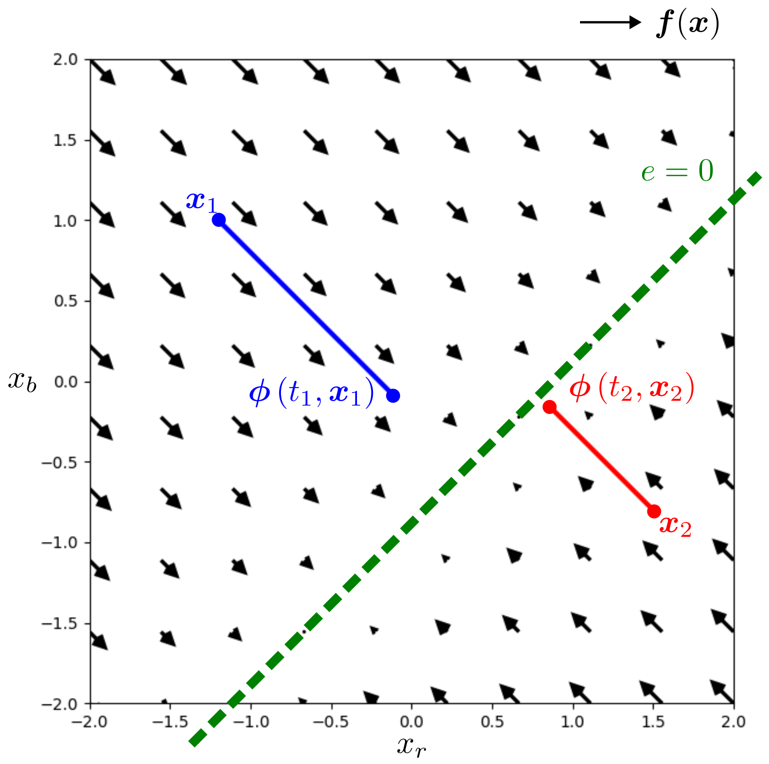


Figure 3.4: Possible trajectories of the continuous-time dynamical system

3.2.1.1 The variational equation

When one has no analytical expression for $\boldsymbol{\phi}$, there exist some numerical method to evaluate or approximate the value of $\boldsymbol{\phi}(t, \boldsymbol{x}_0)$. Some of them will be presented in Section 4.5. These methods often involve the use of the Jacobian of the flow $\boldsymbol{\phi}$ written

$$\boldsymbol{J}_{\phi}(t, \boldsymbol{x}_0) = \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_0}(t, \boldsymbol{x}_0), \forall t \in \mathbb{R}, \forall \boldsymbol{x}_0 \in \mathbb{R}^n, \tag{3.11}$$

and illustrated in Figure 3.5. To make sure this derivative exists, $\boldsymbol{f}$ is now considered to be continuously differentiable such that

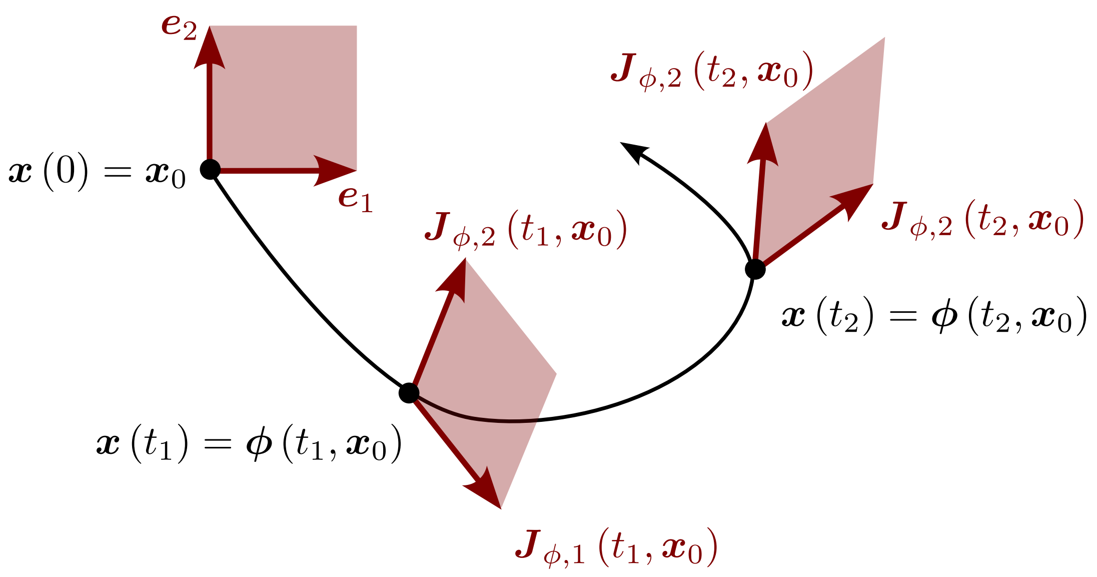
$$\boldsymbol{f} \in \mathcal{C}^1(\mathbb{R}^n), \tag{3.12}$$

making the flow $\boldsymbol{\phi}$ also $\mathcal{C}^1$.

The Jacobian $\boldsymbol{J}_{\phi}$ is often computed via an IVP problem, using the *variational equation* presented in Definition 3.4 and the Theorem 3.1.

Definition 3.4. (Variational equation). Consider the ODE $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$ where $\boldsymbol{f} \in \mathcal{C}^1(\mathbb{R}^n)$. Let $\boldsymbol{x}(t)$ be a solution of this ODE. Then, the *variational equation* along the solution $\boldsymbol{x}(t)$ is defined by

$$\dot{\boldsymbol{J}}(t) = \boldsymbol{A}(t) \cdot \boldsymbol{J}(t), \tag{3.13}$$



$\boldsymbol{e}_2$
$\boldsymbol{x}(0) = \boldsymbol{x}_0$
$\boldsymbol{e}_1$
$\boldsymbol{J}_{\phi,2}(t_2, \boldsymbol{x}_0)$
$\boldsymbol{J}_{\phi,2}(t_1, \boldsymbol{x}_0)$
$\boldsymbol{J}_{\phi,2}(t_2, \boldsymbol{x}_0)$
$\boldsymbol{x}(t_2) = \boldsymbol{\phi}(t_2, \boldsymbol{x}_0)$
$\boldsymbol{x}(t_1) = \boldsymbol{\phi}(t_1, \boldsymbol{x}_0)$
$\boldsymbol{J}_{\phi,1}(t_1, \boldsymbol{x}_0)$

Figure 3.5: Illustration of the flow's Jacobian in two dimensions. The vectors $\boldsymbol{J}_{\phi,1}$ and $\boldsymbol{J}_{\phi,2}$ are the first and second columns of $\boldsymbol{J}_{\phi}$. At the initial time, $\boldsymbol{J}_{\phi,1}(0, \boldsymbol{x}_0) = \boldsymbol{e}_1$ and $\boldsymbol{J}_{\phi,2}(0, \boldsymbol{x}_0) = \boldsymbol{e}_2$ with the Cartesian base $(\boldsymbol{e}_1, \boldsymbol{e}_2)$.

where $\boldsymbol{J}(t)$ is a real matrix and $\boldsymbol{A}(t) = \frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{x}}(\boldsymbol{x}(t))$ is the Jacobian of $\boldsymbol{f}$ evaluated at $\boldsymbol{x}(t)$.

Note that the variational can also be defined for a vector $\boldsymbol{u}(t)$ such that

$$\dot{\boldsymbol{u}}(t) = \boldsymbol{A}(t) \cdot \boldsymbol{u}(t)\,. \tag{3.14}$$

Theorem 3.1. *[37, Section 7] Consider the ODE $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$ where $\boldsymbol{f} \in \mathcal{C}^1(\mathbb{R}^n)$. Let $\boldsymbol{x}(t)$ be a solution of the IVP with $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$. Let $\boldsymbol{U}(t)$ be the solution to the variational equation (3.13) along $x(t)$ that satisfies $\boldsymbol{U}(0) = \boldsymbol{J}_0$. Then*

$$\boldsymbol{J}(t) = \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_0}(t, \boldsymbol{x}_0) \cdot \boldsymbol{J}_0. \tag{3.15}$$

Thus, with $\boldsymbol{J}_0 = \boldsymbol{I}_n$, the solution of the variational equation is $\boldsymbol{J}(t) = \boldsymbol{J}_{\phi}(t, \boldsymbol{x}_0)$. In other words, computing $\boldsymbol{J}_{\phi}$ for a given initial state $\boldsymbol{x}_0$ at the certain time t comes down to integrating the associated variational equation for a duration t.

3.2.2 Discrete dynamical systems

While continuous-time dynamical systems model continuous phenomenons, discrete-time dynamical systems are used to model punctual phenomena, at specific moments in time. Alternatively, they are also used to approximate continuous-time dynamical system to implement them in numerical simulations. In the case of robotics, the embedded computer of the robot is a discrete-time dynamical system. Here is a common definition for discrete-time dynamical systems.

Definition 3.5. (Discrete-time dynamical system) A discrete-time dynamical system is a function $\boldsymbol{\phi}_d : \mathbb{N} \times \mathbb{R}^m \to \mathbb{R}^m$ such that

- for any $\boldsymbol{m} \in \mathbb{R}^m$, $\boldsymbol{\phi}_d(0, \boldsymbol{m}) = \boldsymbol{m}$
- for any $(p, q) \in \mathbb{N}$ and for any $\boldsymbol{m} \in \mathbb{R}^m$, $\boldsymbol{\phi}_d(p, \boldsymbol{\phi}(q, \boldsymbol{m})) = \boldsymbol{\phi}_d(p + q, \boldsymbol{m})$

The discrete-time state is written $\boldsymbol{m}$ as 'memory' to differentiate it from the continuous-time state $\boldsymbol{x}$. Again, for the sake of simplicity, a discrete-time dynamical system may sometimes be referred to as a discrete system. Given an initial vector $\boldsymbol{m}_0 \in \mathbb{R}^m$, the evolution of the discrete dynamical system is described by the sequence $(\boldsymbol{m}_k)_{k\in\mathbb{N}}$ such that $\boldsymbol{m}_k = \boldsymbol{\phi}_d(k, \boldsymbol{m}_0)$ for any $k \in \mathbb{N}$. This sequence is often described in practice by its recursive formula

$$\boldsymbol{m}_{k+1} = \boldsymbol{h}(\boldsymbol{m}_k), \tag{3.16}$$

such that

$$\forall \boldsymbol{m} \in \mathbb{R}^n, \boldsymbol{h}(\boldsymbol{m}) = \boldsymbol{\phi}_d(1, \boldsymbol{m}). \tag{3.17}$$

The mapping is written $\boldsymbol{h}$ to differentiate it from the continuous-time mapping $\boldsymbol{f}$. Thus, compared to the continuous-time dynamical system, the computation of $\boldsymbol{\phi}_d(k, \boldsymbol{m}_0)$ is simple.

Example 3.2. Let us discretise the continuous dynamical system of Example 3.1 with an Euler Scheme to obtain a discrete-time dynamical system that can be used to simulate the system. With a small discretisation time $T > 0$, assume that the vector $\boldsymbol{m}_k$ correspond to the state at the time $t_k = k \cdot T$. With an Euler scheme, the recursive formula of the system is

$$\begin{aligned} \boldsymbol{m}_{k+1} &= \boldsymbol{h}(\boldsymbol{m}_k) \\ &= \boldsymbol{m}_k + T \cdot \boldsymbol{f}(\boldsymbol{m}_k) \end{aligned} \tag{3.18}$$

with $\boldsymbol{f}$ from (3.10). Given an initial condition $\boldsymbol{m}_0$ and a number of iteration $k \in \mathbb{N}$, one can compute discrete time trajectories, as illustrated by Figure 3.6. With a small T, the behaviour of the system is visually similar in continuous time and in discrete time.

3.2.3 Hybrid systems

A hybrid system has both continuous-time and discrete-time behaviour [65]. Depending on the interaction between the continuous and the discrete, the hybrid system can become very complex to describe.

Moreover, the notion of hybrid systems can be associated with the notion of *cyber-physical* systems[100, 102, 116]. Cyber-physical means that there is an interaction

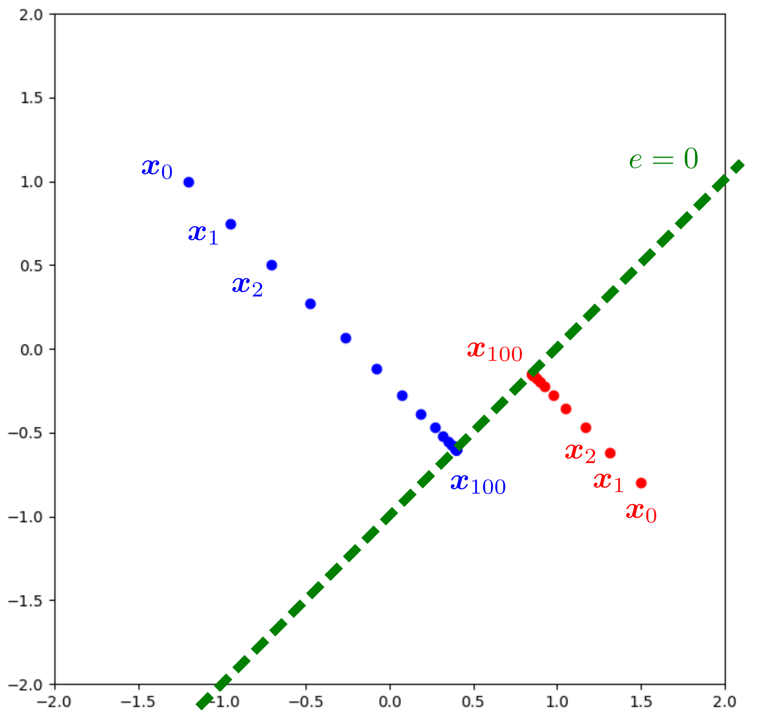


Figure 3.6: Possible trajectories of the discrete-time dynamical system

between computational software and a physical object. But from a control theory perspective, a cyber-physical object is a hybrid as well. These are two different paradigms in different fields, which apply to robots.

A general definition of a hybrid system can be found in [29] and in [30], here presented in Definition 3.6.

Definition 3.6. (Hybrid dynamical system) [30] A hybrid dynamical system is a sextuple

$$\mathcal{H} = (\mathcal{Q}, \mathcal{D}, \boldsymbol{f}, \mathcal{E}, \boldsymbol{g}, \boldsymbol{h}), \tag{3.19}$$

where:

$\mathcal{Q}$ is a **set of modes**, which in most situations can be identified with a subset of the integers,

$\mathcal{D}$ is a **domain map**, which gives, for each $q \in \mathcal{Q}$, the set $D_q \in \mathcal{D}$ in which the continuous state $\boldsymbol{z}$ evolves,

$\boldsymbol{f} : \mathcal{Q} \times \mathbb{R}^p \to \mathbb{R}^p$ is a **flow map**, which describes, through the differential equation,

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{f}(\boldsymbol{z}(t)), \tag{3.20}$$

the continuous evolution of the continuous state variable $\boldsymbol{z}$,

$\mathcal{E} \subset \mathcal{Q} \times \mathcal{Q}$ is a **set of edges**, which identifies the pairs (q, q') such that a transition from the mode q to the mode q' is possible,

$\boldsymbol{g} : \mathcal{E} \to \mathbb{R}^p$ is a **guard map**, which identifies, for each edge $(q, q') \in \mathcal{E}$, the set $\boldsymbol{g}(q, q') \subset \mathbb{R}^p$ to with the continuous state $\boldsymbol{z}$ must belong so that a transition from q to q' occur,

$\boldsymbol{h} : \mathcal{E} \times \mathbb{R}^p \to \mathbb{R}^p$ is a **reset map**, which describes, for each edge $(q, q') \in \mathcal{E}$, the value to which the continuous state $\boldsymbol{z} \in \mathbb{R}^p$ is set during a transition from mode q to mode q'.

This thesis will only cover hybrid systems where discrete behaviours happens at deterministic times. In other words, whatever the initial conditions, a mode transition always occurs at the same time. These systems can be called **synchronous hybrid systems** and are presented by Definition 3.7. The extension of the thesis contributions to all hybrid systems will be the subject of future studies.

Definition 3.7. (synchronous hybrid system) A synchronous hybrid dynamical system is a hybrid system $\mathcal{H} = (\mathcal{Q}, \mathcal{D}, \boldsymbol{f}, \mathcal{E}, \boldsymbol{g}, \boldsymbol{h})$ where the time t is one of the components of the continuous time vector $\boldsymbol{z}$ and where for each edges (q, q'), there is a set of times $\mathcal{T}_{q'} \subset \mathbb{R}$ such that the guard of (q, q') is

$$\boldsymbol{g}(q, q') = \{\boldsymbol{z} \in \mathbb{R} | t \in \mathcal{T}_{q'}\} . \tag{3.21}$$

For convenience, a synchronous hybrid system can be represented such that:

- the time t is separated from the rest of the state vector $\boldsymbol{z}$,
- $\mathcal{Q} = \{1, 2, \dots, N\}$ is identified with N integers. The mode 1 can also be considered mode N+1 and the mode N can also be considered mode 0.
- the only possible edges are $(q, q+1)$ for all $q \in \mathcal{Q}$ and all edge transitions are periodic with a period $T > 0$.

Thus there is a list of increasing time $\{\tau_1, \tau_2, \dots \tau_N\} \in [0, T[$ such that for all $q \in \mathcal{Q}$ and all $k \in \mathbb{Z}$, at time

$$t_{q,k} = \tau_q + T \cdot k, \tag{3.22}$$

the transition $(q-1, q)$ happens, i.e.

$$\forall q \in \mathcal{Q},\, \mathcal{T}_q = \tau_q + T \cdot \mathbb{N}. \tag{3.23}$$

For convenience, $\tau_1 = 0$. One can also represent the time with one indent number such that for all $j \in \mathbb{N}$

$$\begin{aligned} t_j &= \tau_{j - \lfloor \frac{j-1}{N} \rfloor \cdot N} + \left\lfloor \frac{j-1}{N} \right\rfloor \cdot T \\ &= t_{(j - \lfloor \frac{j-1}{N} \rfloor \cdot N), (\lfloor \frac{j-1}{N} \rfloor)}. \end{aligned} \tag{3.24}$$

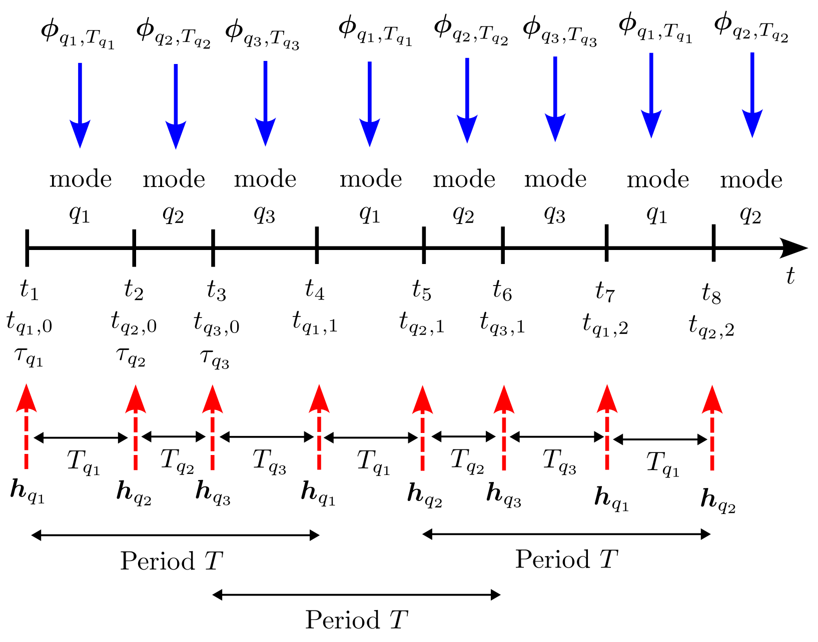


Figure 3.7: Time representation of a synchronous hybrid system with $N = 3$

As a result, each mode $q \in \mathcal{Q}$ last for a duration

$$T_q = (t_{q+1} - t_q) > 0, \tag{3.25}$$

such that $\sum_{q\in\mathcal{Q}} T_q = T$. This time representation is illustrated by Figure 3.7.

Therefore, a synchronous hybrid system can be represented by

$$\begin{cases} \dot{\boldsymbol{z}}(t) & = \boldsymbol{f}_q(\boldsymbol{z}(t)), \text{ if } t \in]t_{q,k}, t_{q,k} + T_q[, \\ \boldsymbol{z}\left(t_{q,k}^+\right) & = \boldsymbol{h}_q(\boldsymbol{z}(t_{q,k})), \end{cases} \tag{3.26}$$

where $t_{q,k}^+$ represent the time instant just after the discrete update at time $t_{q,k}$, and such that for all $q \in \mathcal{Q}$, $\boldsymbol{f}_q(\boldsymbol{z}) = \boldsymbol{f}(q, z)$ and $\boldsymbol{h}_q(\boldsymbol{z}) = \boldsymbol{h}(q, \boldsymbol{z})$. As a result, the synchronous hybrid system has a cycle mapping

$$\boldsymbol{h}_{\text{cycle}} = \boldsymbol{\phi}_{N,T_N} \circ \boldsymbol{h}_N \circ \ldots \circ \boldsymbol{\phi}_{2,T_2} \circ \boldsymbol{h}_2 \circ \boldsymbol{\phi}_{1,T_2} \circ \boldsymbol{h}_1$$

where $\boldsymbol{\phi}_{q,T_q}$ is the flow of $\boldsymbol{f}_q$ for a duration T_q, as illustrated by Figure 3.8.

Example 3.3. To have a practical illustration of a synchronous hybrid system, consider the robots of the example 3.1 with the continuous-time dynamical system (3.10). Assume that the positions of the robots are now periodically measured with the period $T > 0$. These measurements create a cycle composed of two discrete-time events, illustrated by Figure 3.9. For every $k \in \mathbb{N}$, the position of *Red* $x_r(t_{r,k})$ is measured at the time $t_{r,k} = k \cdot T$, and, the position of Blue *Blue* $x_b(t_{b,k})$ is measured at the time $t_{b,k} = k \cdot T + \frac{T}{2}$. The cycle of measurement is repeated indefinitely.

Thus, this system can be represented with two modes $q_r = 1$ and $q_b = 2$ with two possible edges (q_r, q_b) and (q_b, q_r). The transition from q_r to q_b happens on the times $t_{b,k}$. The transition from q_b to q_r happens on the times $t_{r,k}$.

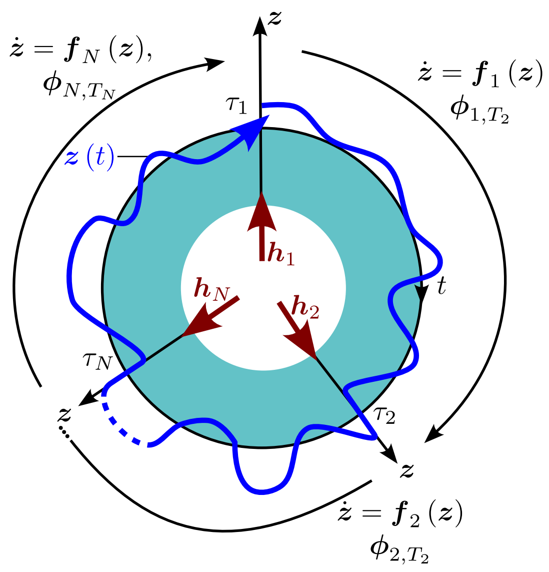

$\dot{\boldsymbol{z}} = \boldsymbol{f}_N(\boldsymbol{z})$, $\boldsymbol{\phi}_{N,T_N}$
$\boldsymbol{z}$
$\dot{\boldsymbol{z}} = \boldsymbol{f}_1(\boldsymbol{z})$ $\boldsymbol{\phi}_{1,T_2}$
τ_1
$\boldsymbol{z}(t)$
$\boldsymbol{h}_1$
$\boldsymbol{h}_N$
$\boldsymbol{h}_2$
t
τ_N
τ_2
$\boldsymbol{z}$
$\boldsymbol{z}$
$\dot{\boldsymbol{z}} = \boldsymbol{f}_2(\boldsymbol{z})$ $\boldsymbol{\phi}_{2,T_2}$

Figure 3.8: Illustration of the cycle of a synchronous hybrid system.

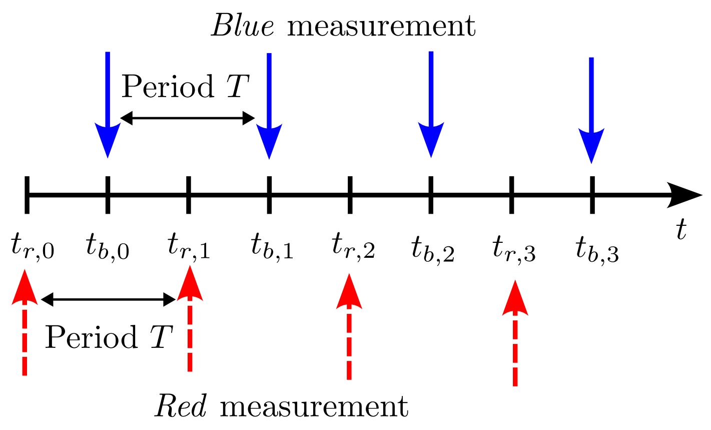

Blue measurement
Period T
$t_{r,0}$ $t_{b,0}$ $t_{r,1}$ $t_{b,1}$ $t_{r,2}$ $t_{b,2}$ $t_{r,3}$ $t_{b,3}$
t
Period T
Red measurement

Figure 3.9: Clock and discrete-time events of the system

Then, the measurements are memorised by the robots in the shared memory variables

$$\begin{aligned} m_{r,k} &= x_r\left(t_{r,k}\right), \forall t \in \left[t_{r,k}, t_{r,k+1}\right[, \\ m_{b,k} &= x_b\left(t_{b,k}\right), \forall t \in \left[t_{b,k}, t_{b,k+1}\right[. \end{aligned}$$

Moreover, at any time, the speed of the robot is set as

$$\begin{bmatrix} \dot{x}_r(t) \\ \dot{x}_b(t) \end{bmatrix} = \arctan\left(\left(m_r(t) - m_b(t) - d\right)\right) \begin{bmatrix} -1 \\ 1 \end{bmatrix}. \tag{3.27}$$

Then, consider the continuous state vector

$$\boldsymbol{z}(t) = \begin{bmatrix} x_r(t) & x_b(t) & m_r(t) & m_b(t) \end{bmatrix}^T \in \mathbb{R}^4. \tag{3.28}$$

From (3.27) the state $\boldsymbol{z}$ is subject to the vector field

$$\begin{aligned} \boldsymbol{f}_{rb} : \mathbb{R}^4 &\to \mathbb{R}^4, \\ \boldsymbol{z} &\mapsto \begin{bmatrix} -\arctan\left(k_p \cdot \left(m_r - m_b - d\right)\right) \\ \arctan\left(k_p \cdot \left(m_r - m_b - d\right)\right) \\ 0 \\ 0 \end{bmatrix}, \end{aligned} \tag{3.29}$$

that describes the continuous-time evolution of the system such that

$$\dot{\boldsymbol{z}} = \boldsymbol{f}_{rb}(\boldsymbol{z}). \tag{3.30}$$

Moreover, the state $\boldsymbol{z}$ is subject to the two measurement events. At time $t_{r,k}$, the state $\boldsymbol{z}$ is updated with the function $\boldsymbol{h}_r$ given by

$$\begin{aligned} \boldsymbol{h}_r :& \mathbb{R}^4 \to \mathbb{R}^4, \\ & \boldsymbol{z} \mapsto \begin{bmatrix} x_r & x_b & x_r & m_b \end{bmatrix}^T \end{aligned} \tag{3.31}$$

such that

$$\boldsymbol{z}\left(t_{r,k}^+\right) = \boldsymbol{h}_r\left(\boldsymbol{z}\left(t_{r,k}\right)\right). \tag{3.32}$$

The time $t_{r,k}^+$ is the instant just after the update perform at instant $t_{r,k}$. In the same way, at time $t_{b,k}$, the state $\boldsymbol{z}$ is updated as

$$\boldsymbol{z}\left(t_{b,k}^+\right) = \boldsymbol{h}_b\left(\boldsymbol{z}\left(t_{b,k}\right)\right), \tag{3.33}$$

with the function $\boldsymbol{h}_b$ given by

$$\begin{aligned} \boldsymbol{h}_b :& \mathbb{R}^4 \to \mathbb{R}^4, \\ & \boldsymbol{z} \mapsto \begin{bmatrix} x_r & x_b & m_r & x_b \end{bmatrix}^T. \end{aligned} \tag{3.34}$$

Some trajectories of the system are illustrated by Figure 3.10. As illustrated by Figure 3.11, given an initial condition $\boldsymbol{z}(0)$, the evolution of the state $\boldsymbol{z}$ is described by a composition of with the flow $\boldsymbol{\phi}_{\frac{T}{2}}$ of $\boldsymbol{f}_{rb}$ for a time $\frac{T}{2}$ and with the discrete-time updates functions $\boldsymbol{h}_r$ and $\boldsymbol{h}_b$.

Note that in this example, the state vector $\boldsymbol{z}$ is split to differentiate the physical variables of the system $\boldsymbol{x}(t) = [x_r(t), x_b(t)]$ and the memory variables $m_k = [m_r(t_{2k}), m_b(t_{2k+1})]$ that only update with the discrete-time event.

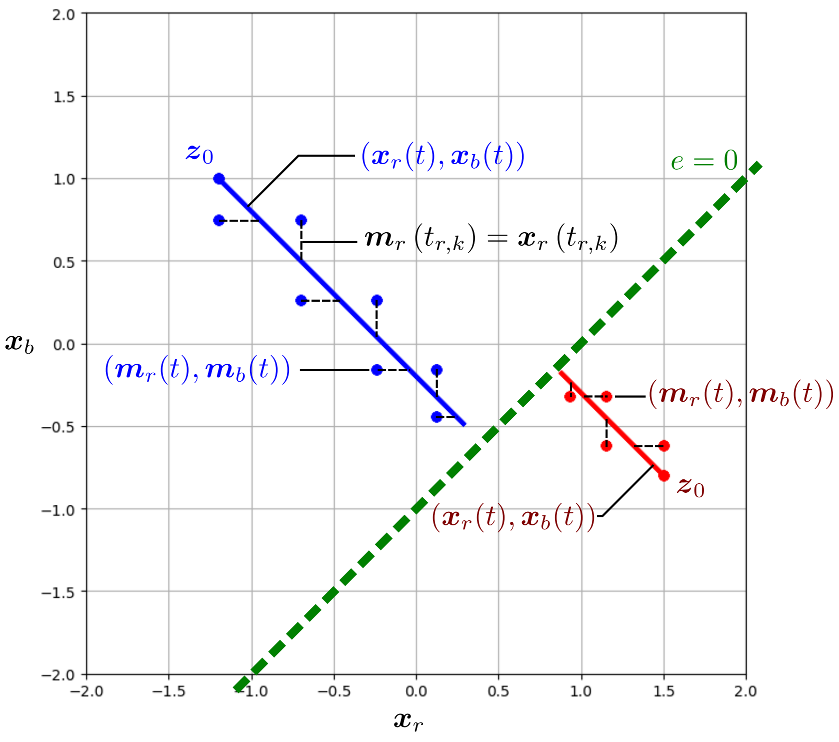


Figure 3.10: Possible trajectories of the synchronous hybrid dynamical system

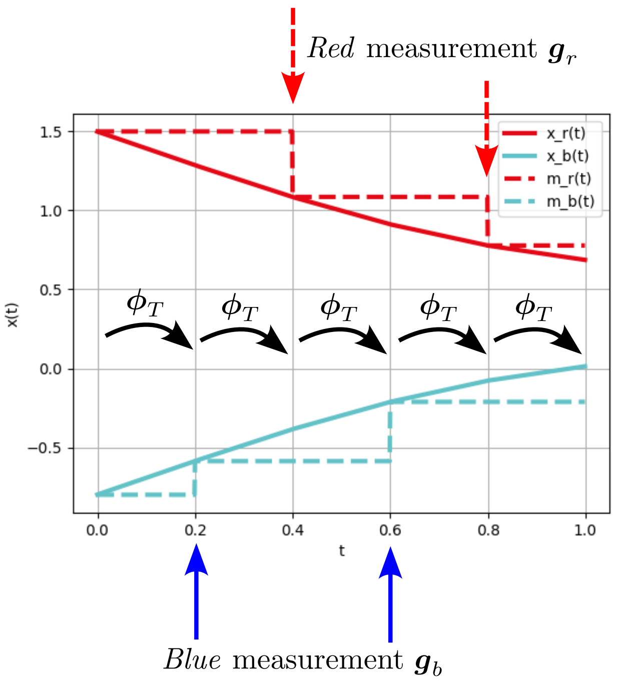


Figure 3.11: Time evolution of the state of the system. The time evolution of a synchronous hybrid system is cyclic

3.2.4 Lipschitz functions

In Chapter 5 and 8, some mappings will be considered K-Lipschitz, as presented in Definition 3.9. This definition relies on the notion of metric space, presented in Definition 3.8. As presented by Theorem 3.2, Lipschitz functions are also $\mathcal{C}^1$ with bounded derivative.

Definition 3.8. (Metric space) Let $\mathcal{X}$ be a non-empty set. A mapping $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a metric on $\mathcal{X}$ if for all $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathcal{X}$, one verifies

$$d(\boldsymbol{x}, \boldsymbol{y}) = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{y} \,(\text{separation}), \tag{3.35}$$

$$d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x}) \,(\text{symmetry}), \tag{3.36}$$

$$\forall \boldsymbol{z} \in \mathcal{X},\; d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z}) + d(\boldsymbol{y}, \boldsymbol{z}) \,(\text{triangle inequality}). \tag{3.37}$$

Given a matrix d on a set $\mathcal{X}$, the paid $(\mathcal{X}, d)$ is a metric space.

Definition 3.9. (K-Lipschitz). Let $K > 0$ and consider the metric spaces $(\mathbb{R}^n, d_n)$ and $(\mathbb{R}^m, d_m)$. The mapping $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ is K-Lipschitz if and only if for all $(\boldsymbol{x}_a, \boldsymbol{x}_b) \in \mathbb{R}^p$ one has

$$d_m(\boldsymbol{f}(\boldsymbol{x}_a), \boldsymbol{f}(\boldsymbol{x}_b)) \leq K \cdot d_n(\boldsymbol{x}_a, \boldsymbol{x}_b). \tag{3.38}$$

Any finite-dimensional vector space $\mathcal{X}$ with a norm $\|.\|$ is a metric space with a metric

$$\begin{aligned} d_{\|.\|} : \mathbb{R}^n \times \mathbb{R}^n &\to \mathbb{R}, \\ \boldsymbol{x}, \boldsymbol{y} &\mapsto \|\boldsymbol{x} - \boldsymbol{y}\|. \end{aligned} \tag{3.39}$$

Theorem 3.2. *Let $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ be K-Lipschitz. Then $\boldsymbol{f}$ is $\mathcal{C}^1$.*

When the function $\boldsymbol{f}$ of a continuous-time dynamical system is K-Lipschitz, then it verifies Theorem 3.3.

Theorem 3.3. *[37, Chapter 17.3] Let $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ be Lipschitz over the metric space $\left(\mathbb{R}^n, d_{\|.\|}\right)$ with a constant K. Let $\boldsymbol{x}_a(t)$ and $\boldsymbol{x}_b(t)$ be solutions of the ODE*

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t)), \tag{3.40}$$

on the time interval $[t_0, t_f]$. Then, for all $t \in [t_0, t_f]$, one has

$$d_{\|.\|}(\boldsymbol{x}_a(t), \boldsymbol{x}_b(t)) \leq d_{\|.\|}(\boldsymbol{x}_a(t_0), \boldsymbol{x}_b(t_0)) \cdot e^{K(t-t_0)}. \tag{3.41}$$

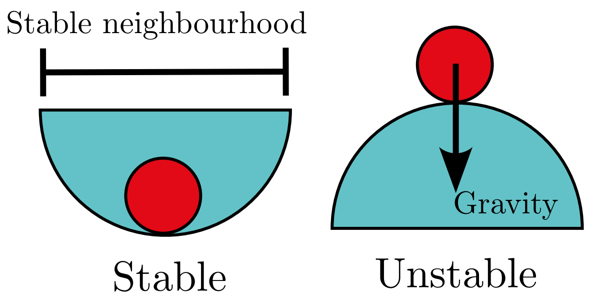


Figure 3.12: Illustration of stable and unstable equilibrium points, with the notion of stable neighbourhood.

3.3 Stability analysis

As presented in Chapter 1, it is important for the formation control to show some form of stability to make sure that the robots will stay in formation. Mathematically speaking, this problem consists in proving that the formation is a stable equilibrium point for the dynamical system that models the group of robots. An *equilibrium points* of a dynamical system is a constant trajectory. It is defined as stable if the nearby trajectories of the system stay nearby after some time.

Figure 3.12 is a common illustration of equilibrium points. In this example, a marble is subject to gravity. When the marble is at the centre of an upright bowl or an upside-down bowl, it remains still: the centre position is an equilibrium point. When the marble is moved away from the centre of the upright bowl, it will come back to the centre. However, when the marble is moved away from the centre of the upside-down bowl, it will diverge away. Thus, the equilibrium point is stable when the bowl is upright and unstable when the bowl is upside down.

Real systems are often only stable in a neighbourhood of the equilibrium point. For example, if the marble exits the upright bowl, it won't come back. As a result, describing the size and shape of the stable neighbourhood is also a common topic in stability analysis.

This Section introduces some background on the stability of equilibrium points for dynamical systems, based on the Lyapunov Theory, as presented in the books [44, Chapter 4] and [37] and in [7].

3.3.1 Stability of nonlinear continuous dynamical systems

The Lyapunov stability is a very common stability analysis method for nonlinear dynamical systems. It was introduced by Lyapunov in 1892 [64]. The main interest of the Lyapunov stability is that it studies the system's stability without solving nonlinear ODE. In this Section, consider the nonlinear continuous dynamical system given by the ODE

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t)), \tag{3.42}$$

as in Section 3.2.1. Assume that $\bar{\boldsymbol{x}}$ is an *equilibrium point* of the system, such that $\boldsymbol{f}(\bar{\boldsymbol{x}}) = 0$, making the trajectory $\boldsymbol{x}(t) = \bar{\boldsymbol{x}}$ a constant solution of the ODE.

Remark 3.1. For convenience, and without loss of generality, the equilibrium point is considered at the origin of $\mathbb{R}^n$ such that $\bar{\boldsymbol{x}} = 0$. Even when the equilibrium point is not the origin $\bar{\boldsymbol{x}} \neq 0$, it can be shifted by the change of variable $\boldsymbol{y} = \boldsymbol{x} - \bar{\boldsymbol{x}}$. The new variable $\boldsymbol{y}$ is solution to the ODE

$$\begin{aligned}\dot{\boldsymbol{y}}(t) &= \dot{\boldsymbol{x}}(t), \\ &= \boldsymbol{f}(\boldsymbol{x}(t)) \\ &= \boldsymbol{f}(\boldsymbol{y}(t) + \bar{\boldsymbol{x}}) \\ &= \boldsymbol{g}(\boldsymbol{y}(t)) \end{aligned} \tag{3.43}$$

with $\boldsymbol{g}(\boldsymbol{y}) = \boldsymbol{f}(\boldsymbol{y} + \bar{\boldsymbol{x}})$. The equilibrium point of the ODE (3.43) is $\boldsymbol{y}_e = 0$. Since the ODE (3.42) and (3.43) represent the same system, it is more convenient to choose the representation with the equilibrium point at the origin.

3.3.1.1 Stability definition

The stability of a system is illustrated by Figure 3.13 and defined by the following definition.

Definition 3.10. Let $\bar{\boldsymbol{x}} = 0$ be an equilibrium point for the system of equation (3.42). For this system, the equilibrium point is:

- *Stable,* if for every $\epsilon > 0$, there exists $\delta > 0$ such that

$$\|\boldsymbol{x}(0)\| < \delta \Rightarrow \|\boldsymbol{x}(t)\| < \epsilon, \forall t \geq 0 \tag{3.44}$$

- *Unstable*, if it is not stable.
- *Attractive*, if there exists $\delta > 0$ such that

$$\|\boldsymbol{x}(0)\| < \delta \Rightarrow \lim_{t \to \infty} \|\boldsymbol{x}(t)\| = 0 \tag{3.45}$$

- *Asymptotically stable,* if it is stable and attractive.
- *Exponentially stable,* if it is asymptotically stable and there exist $\alpha > 0$, $\beta > 0$, $\delta > 0$ such that

$$\|\boldsymbol{x}(0)\| < \delta \Rightarrow \|\boldsymbol{x}(t)\| \leq \alpha \|\boldsymbol{x}(0)\| e^{-\beta t}, \forall t \geq 0 \tag{3.46}$$

A stable equilibrium point means that if the system is initialised near the equilibrium point, it will stay nearby. An attractive equilibrium point means that if the system is initialised in a specific neighbourhood of the equilibrium point, the state will converge towards the equilibrium point. In addition, exponential stability means that the system's convergence follows an exponential function.

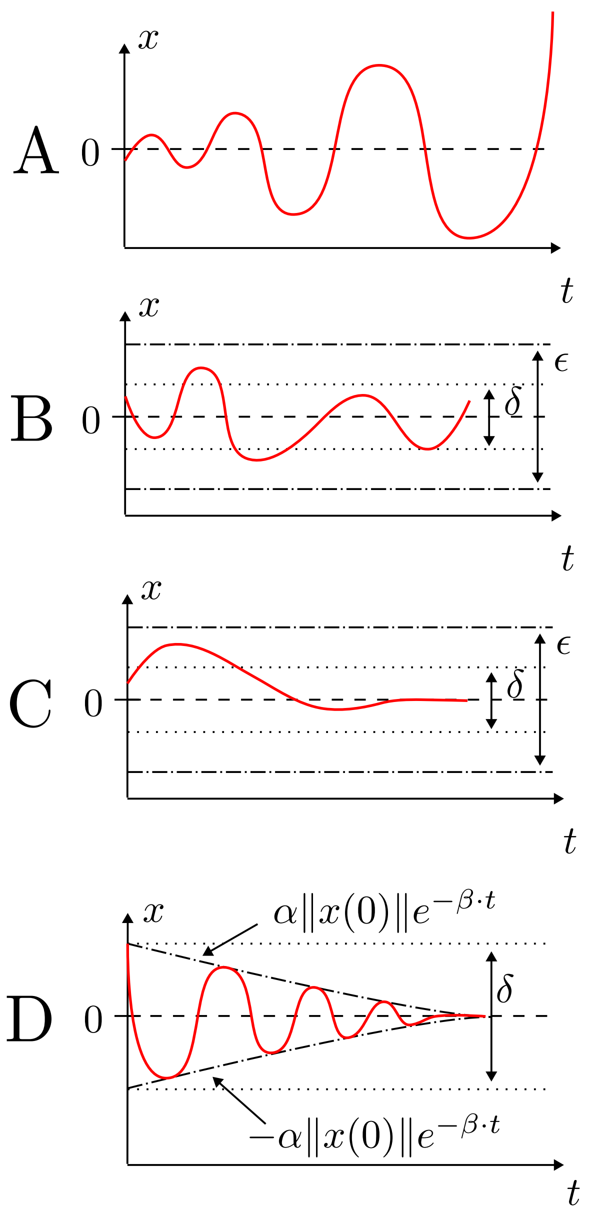


Figure 3.13: Illustration of instability (A), stability (B), asymptotic stability (C) and exponential stability (D) one a one-dimensional system. Trajectories near the equilibrium point x_e remain in a neighbourhood of this point.

3.3.1.2 Local and global asymptotic stability

Many systems are only asymptotically stable up to a certain distance from the equilibrium point. For these systems, one can determine its *region of attraction*

$$\Omega_{att} = \left\{ \boldsymbol{x}_0 \in \mathbb{R}^n | \lim_{t\to\infty} \phi\left(t, \boldsymbol{x}_0\right) = \bar{\boldsymbol{x}} \right\} \tag{3.47}$$

where ϕ is the flow of the ODE (3.42). When $\Omega_{\text{att}} = \mathbb{R}^n$, the asymptotic stability is *global.* When $\Omega_{\text{att}} \subset \mathbb{R}^n$, the asymptotic stability is *local.* Note that when no analytical expression for ϕ is known, it is hard to describe Ω_{att} analytically.

3.3.1.3 Stability analysis by linearisation

In some cases, stability can be proved by linearising the system and by using the Theorem 3.4.

Theorem 3.4. *Consider the nonlinear system*

$$\dot{\boldsymbol{x}}\left(t\right) = \boldsymbol{f}\left(\boldsymbol{x}\left(t\right)\right), \tag{3.48}$$

where $\boldsymbol{f} : \mathcal{D} \to \mathbb{R}^n$ is continuously differentiable on the domain of definition $\mathcal{D} \subseteq \mathbb{R}^n$, and where the origin $\bar{\boldsymbol{x}} = 0$ is an equilibrium point of the system. This system can be linearised at the origin into the LTI (linear time-invariant) system

$$\dot{\boldsymbol{x}}\left(t\right) = \boldsymbol{A} \cdot \boldsymbol{x}\left(t\right), \tag{3.49}$$

with the Jacobian matrix

$$\boldsymbol{A} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\left(0\right). \tag{3.50}$$

The equilibrium point $\bar{\boldsymbol{x}} = 0$ of the nonlinear dynamical system (3.48) is locally asymptotically stable if $Re\left(\lambda_i\right) < 0$ for all eigenvalues of $\boldsymbol{A}$ (i.e. $\boldsymbol{A}$ is Hurwitz). The equilibrium point is unstable if $Re\left(\lambda_i\right) > 0$ for one or more of the eigenvalues of $\boldsymbol{A}$.

However, this Theorem has several practical limits:

1. When the eigenvalues of $\boldsymbol{A}$ have negative real part but some eigenvalues have a null real part $\text{Re}\left(\lambda_i\right) = 0$, it is not possible to conclude on the stability of the equilibrium point for the nonlinear system. Thus, while linearisation is a simple and fast method to study stability, it can not study all systems.

2. This theorem only proves local stability and gives no information on the region of attraction.

3. Eigenvalues are usually computed by computers. However, it is challenging to find an algorithm that guarantees the computation of the eigenvalues.

3.3.1.4 Proving stability with Lyapunov functions

Lyapunov functions are a more powerful tool to study stability. They are used with the following theorem.

Theorem 3.5. *Consider the nonlinear system*

$$\dot{\boldsymbol{x}}\left(t\right) = \boldsymbol{f}\left(\boldsymbol{x}\left(t\right)\right),$$

with $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$*, such that* $\bar{\boldsymbol{x}} = 0$ *is an equilibrium point of the system. Consider a function* $V : \mathbb{R}^n \to \mathbb{R}$ *such that* $V\left(\bar{\boldsymbol{x}}\right) = 0$ *and* $V\left(\boldsymbol{x}\right) > 0$ *in a neighbourhood* E *of* $\bar{\boldsymbol{x}}$*. If such function exist, then*

- if $\dot{V}\left(\boldsymbol{x}\right) \leq 0$ for all $\boldsymbol{x} \in E$,then $\bar{\boldsymbol{x}}$ is *stable*,
- if $\dot{V}\left(\boldsymbol{x}\right) < 0$ for all $\boldsymbol{x} \in E \setminus \{\bar{\boldsymbol{x}}\}$,then $\bar{\boldsymbol{x}}$ is *asymptotically stable*,
- if $\dot{V}\left(\boldsymbol{x}\right) > 0$ for all $\boldsymbol{x} \in E \setminus \{\bar{\boldsymbol{x}}\}$,then $\bar{\boldsymbol{x}}$ is *unstable.*

Theorem 3.5 shows that if there is a continuously differentiable positive definite function $V\left(\boldsymbol{x}\right)$ such that $\dot{V}\left(\boldsymbol{x}\right)$ is negative semi-definite, then the equilibrium point is stable. In addition, if $\dot{V}\left(\boldsymbol{x}\right)$ is negative definite, the equilibrium point is asymptotically stable.

The function V is called a *Lyapunov function* when $\bar{\boldsymbol{x}}$ is *stable.* The surface $V\left(\boldsymbol{x}\right) = c$ with $c \in \mathbb{R}^+$ is then called a *Lyapunov surface* or a *level surface.* Consider the set

$$\Omega_c = \{\boldsymbol{x} \in \mathbb{R}^n | V\left(x\right) \leq c\} \tag{3.51}$$

whose border is a level surface. As illustrated by Figure 3.14, if V is a Lyapunov function, one has $\dot{V}\left(x\right) \leq 0$, so every trajectory entering in Ω_c will remain in Ω_c. Moreover, on the border of Ω_c, the vector field defined by $\boldsymbol{f}$ points inside Ω_c. Therefore, the set Ω_c is included in the region of attraction and can be used to make an inner approximation of the latter.

From a physical point of view, Lyapunov functions can be seen as a form of energy the system initially has. This energy decreases over time, leading the systems towards the equilibrium point, which is the point of lowest energy. While the energy of a system can be a Lyapunov function, there are other forms of Lyapunov functions.

It is important to know that the Lyapunov functions are only a sufficient condition for stability. When a Lyapunov is found, the equilibrium point is stable. However, failure to find a Lyapunov function does not mean that the equilibrium point is unstable.

There exist several methods to find good candidates for a Lyapunov function. However, researchers are studying more and more complex systems with more variables and more elaborated nonlinear behaviours, making Lyapunov functions more challenging to find. This is particularly the case of formation control where a lot of time and effort is spent on finding these Lyapunov functions, as in [110].

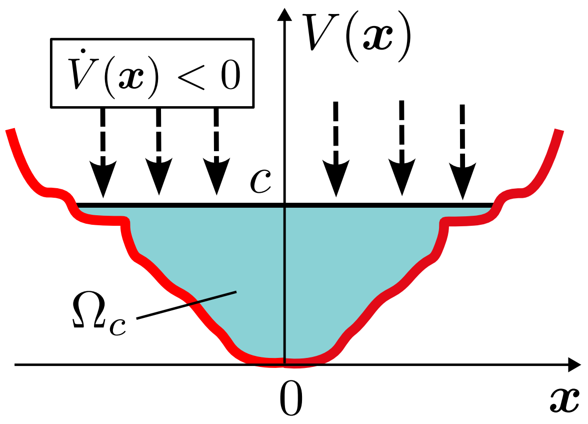


Figure 3.14: Lyapunov function and the level surface

Example 3.4. To illustrate a practical proof with a Lyapunov function, consider the simple pendulum with quadratic friction, represented by the following differential equation

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\sin(x_1) - |x_2| \cdot x_2 \end{bmatrix} \tag{3.52}$$

where x_1 is the angle of the pendulum, x_2 is the angular speed. This system has an equilibrium point at the origin. The mechanical energy of this system is given by

$$E_m = \frac{1}{2}x_2^2 + (1 - \cos(x_1)). \tag{3.53}$$

Then, consider the function

$$\begin{aligned} V :&\mathbb{R}^2 \to \mathbb{R}, \\ &\boldsymbol{x} \mapsto \frac{1}{2}x_2^2 + (1 - \cos(x_1)). \end{aligned} \tag{3.54}$$

One can verify that

$$\begin{cases} V(\boldsymbol{x}) = 0 & \text{if } \boldsymbol{x} \neq \boldsymbol{0}, \\ V(\boldsymbol{x}) > 0 & \text{else.} \end{cases} \tag{3.55}$$

Moreover, the time derivative of V is given by

$$\begin{aligned} \dot{V}(\boldsymbol{x}) &= x_2 \cdot \dot{x}_2 + \sin(x_1) \cdot \dot{x}_1, \\ &= x_2 \cdot (-\sin(x_1) - |x_2| \cdot x_2) + \sin(x_1) \cdot x_2, \\ &= -|x_2| \cdot x_2^2. \end{aligned}$$

So, one can verify that for any $\boldsymbol{x} \in \mathbb{R}^2$, $\dot{V}(\boldsymbol{x}) \leq 0$ and $\dot{V}(x_1, 0) = 0$. Therefore, the Function V is a Lyapunov function of the system for the stable equilibrium point $\bar{\boldsymbol{x}} = 0$. Note that $\dot{V}(x)$ is only negative semi-definite, so one cannot deduce if the equilibrium point is asymptotically stable). Figure 3.15 illustrate some trajectories and some level surfaces.

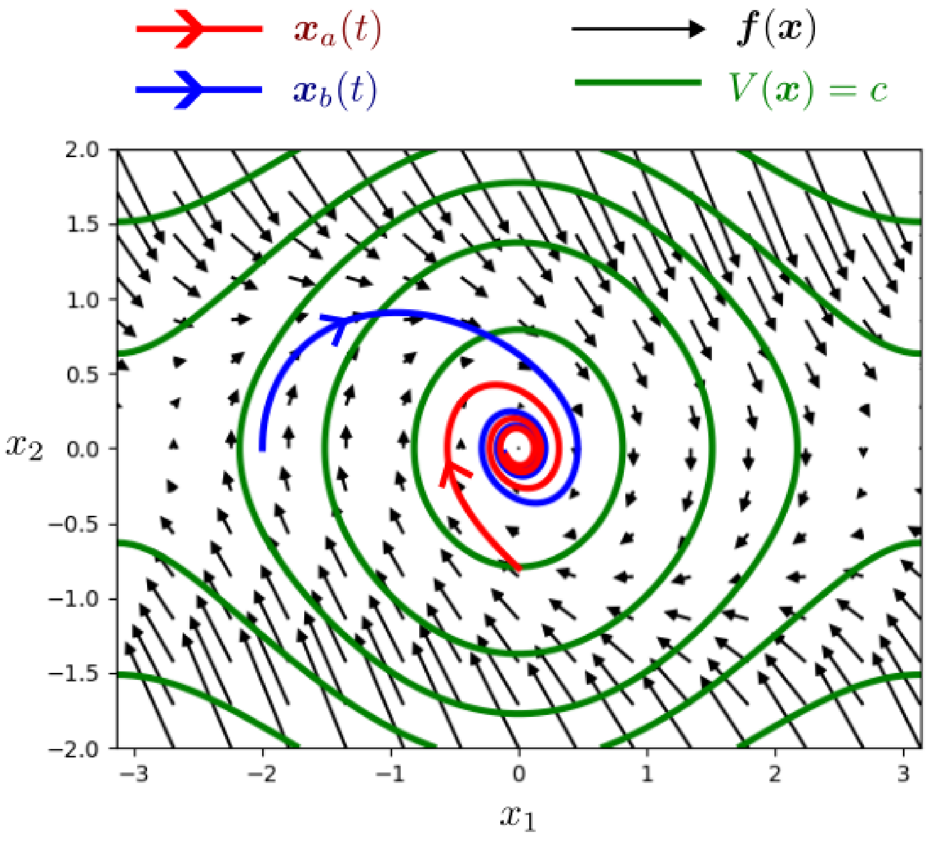


Figure 3.15: Illustration of Example 3.4. The trajectories $\boldsymbol{x}_a(t)$ and $\boldsymbol{x}_b(t)$ converge to the equilibrium point $\bar{\boldsymbol{x}} = 0$, following the vector field $\boldsymbol{f}(\boldsymbol{x})$. Some level surfaces of V are represented. The level c increases when moving away from the equilibrium point. The trajectory crosses the level surfaces in a decreasing order. the curves $V(\boldsymbol{x}) = c$ of the Lyapunov function are all the contours of a Positive invariant inner set.

3.3.1.5 The continuous Lyapunov equation

In the case of a linear time-invariant system like

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A} \cdot \boldsymbol{x}(t), \tag{3.56}$$

Lyapunov functions can have a quadratic form like

$$V(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{P} \boldsymbol{x}, \tag{3.57}$$

with $\boldsymbol{P} \in \mathcal{S}_n^+$. As presented by Theorem 3.6, if $\boldsymbol{P}$ is a solution of a Lyapunov equation, then for all $\boldsymbol{x} \in \mathbb{R}$, one has

$$\begin{aligned}\dot{V}(\boldsymbol{x}) &= \boldsymbol{x}^T \boldsymbol{P} \dot{\boldsymbol{x}} + \dot{\boldsymbol{x}}^T \boldsymbol{P} \boldsymbol{x} \\ &= \boldsymbol{x}^T \left(\boldsymbol{A}^T \boldsymbol{P} + \boldsymbol{P} \boldsymbol{A}\right) \boldsymbol{x}. \\ &\overset{(3.59)}{=} -\boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} \\ &< 0. \end{aligned} \tag{3.58}$$

As it will be presented in Section 4.3, ellipsoids are described by the same quadratic form (3.57). Moreover, the level surface of such Lyapunov functions are ellipsoids. Thus the Lyapunov equation (3.59) will become useful when studying stability with ellipsoids.

Theorem 3.6. *[44, Theorem 4.6]A matrix $\boldsymbol{A}$ is Hurwitz; that is $Re(\lambda_i) < 0$ for all its eigenvalues of $\boldsymbol{A}$ if and only if for any $\boldsymbol{Q} \in \mathcal{S}_n^+$ there exists a $\boldsymbol{P} \in \mathcal{S}_n^+$ that satisfies the Lyapunov equation*

$$\boldsymbol{A}^T \boldsymbol{P} + \boldsymbol{P} \boldsymbol{A} + \boldsymbol{Q} = \boldsymbol{0}. \tag{3.59}$$

Moreover, if $\boldsymbol{A}$ is Hurwitz, then $\boldsymbol{P}$ is the unique solution of (3.59).

3.3.2 Stability of discrete systems

The notion of Lyapunov stability can be extended for discrete-time systems as presented in [7]. Consider the nonlinear discrete-time system given by the recurrence mapping

$$\boldsymbol{m}_{k+1} = \boldsymbol{h}(\boldsymbol{m}_k). \tag{3.60}$$

A state $\bar{\boldsymbol{m}}$ is an *equilibrium point* of the system if $\boldsymbol{h}(\bar{\boldsymbol{m}}) = \bar{\boldsymbol{m}}$. The point $\bar{\boldsymbol{m}}$ is also called a *fixed point* of the function $\boldsymbol{h}$. Again, assume that $\bar{\boldsymbol{m}} = 0$, since one can always apply a change of variable $\boldsymbol{y} = \boldsymbol{m} - \bar{\boldsymbol{m}}$ on the system. The definition of the Lyapunov stability for discrete system is an extension of Definition 3.10.

Definition 3.11. Let $\bar{\boldsymbol{m}} = 0$ be an equilibrium point for the system of equation (3.60). For this system, the equilibrium point is:

- *Stable,* if for every $\epsilon > 0$, there exists $\delta > 0$ such that

$$\|\boldsymbol{m}_0\| < \delta \Rightarrow \|\boldsymbol{m}_k\| < \epsilon, \forall k \in \mathbb{N} \tag{3.61}$$

- *Unstable*, if it is not stable.
- *Attractive*, if there exists $\delta > 0$ such that

$$\|\boldsymbol{m}_0\| < \delta \Rightarrow \lim_{k\to\infty} \|\boldsymbol{m}_k\| = 0 \tag{3.62}$$

- *Asymptotically stable,* if it is stable and attractive.
- *Exponentially stable,* if it is asymptotically stable and there exist $\alpha > 0$, $\beta > 0$, $\delta > 0$ such that

$$\|\boldsymbol{m}_0\| < \delta \Rightarrow \|\boldsymbol{m}_k\| \leq \alpha \|\boldsymbol{m}_0\| e^{-\beta k}, \forall k \in \mathbb{N} \tag{3.63}$$

As in Section 3.3.1.2, the stability can be global or local depending on the size of the region of attraction

$$\Omega_{\text{att}} = \left\{ \boldsymbol{m}_0 \in \mathbb{R}^n | \lim_{k\to\infty} \boldsymbol{h}^k (\boldsymbol{m}_0) = \bar{\boldsymbol{x}} \right\} \tag{3.64}$$

For, the linearising method, Theorem 3.4 is extended to Theorem 3.7.

Theorem 3.7. *Consider the nonlinear discrete system*

$$\boldsymbol{m}_{k+1} = \boldsymbol{h}(\boldsymbol{m}_k),$$

with the equilibrium $\bar{\boldsymbol{m}} = 0$ and $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^n$. Let $\boldsymbol{F} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{m}}(0)$. The equilibrium point $\bar{\boldsymbol{m}}$ of the nonlinear discrete system is locally asymptotically stable if $|\lambda_i| < 1$ for all eigenvalues of $\boldsymbol{F}$. In addition, if $\boldsymbol{h}$ is K-Lipschitz, then the system is locally exponentially stable. The equilibrium point is unstable if $|\lambda_i| > 1$ for one or more of the eigenvalues of $\boldsymbol{F}$.

A matrix $\boldsymbol{F}$ with all the eigenvalues in absolute value smaller than 1 is called a **Schur matrix**. Moreover, if $\boldsymbol{h}$ is Lipschitz, the system is locally exponentially stable. Then, for Lyapunov function, Theorem 3.5 can be extended to Theorem 3.8.

Theorem 3.8. *Consider the system of equation (3.60) with the equilibrium point $\bar{\boldsymbol{m}} = 0$. Consider a function $V : \mathbb{R}^n \to \mathbb{R}$ such that $V(\bar{\boldsymbol{m}}) = 0$ and $V(\boldsymbol{m}) > 0$ in a neighbourhood E of $\bar{\boldsymbol{m}}$. If such a function exists, then*

- if $V(\boldsymbol{h}(\boldsymbol{m})) - V(\boldsymbol{m}) \leq 0$ for all $\boldsymbol{m} \in E$, then $\bar{\boldsymbol{m}}$ is *stable*,
- if $V(\boldsymbol{h}(\boldsymbol{m})) - V(\boldsymbol{m}) < 0$ for all $\boldsymbol{m} \in E \setminus \{\bar{\boldsymbol{x}}\}$, then $\bar{\boldsymbol{m}}$ is *asymptotically stable,*
- if $V(\boldsymbol{h}(\boldsymbol{m})) - V(\boldsymbol{m}) > 0$ for all $\boldsymbol{m} \in E \setminus \{\bar{\boldsymbol{x}}\}$, then $\bar{\boldsymbol{m}}$ is *unstable.*

Finally, there is also a *discrete Lyapunov equation* given by

$$\boldsymbol{F}^T \boldsymbol{P} \boldsymbol{F} + \boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0}, \tag{3.65}$$

where $\boldsymbol{P}$ and $\boldsymbol{Q}$ are positive definite matrix. This equation is deduced from the Lyapunov quadratic function $V(\boldsymbol{m}) = \boldsymbol{m}^T \boldsymbol{Q} \boldsymbol{m}$ and the linear discrete mapping $\boldsymbol{m}_{k+1} = \boldsymbol{h}(\boldsymbol{m}_k) = \boldsymbol{F} \cdot \boldsymbol{m}_k$ which give

$$\begin{aligned} V(\boldsymbol{h}(\boldsymbol{m})) - V(\boldsymbol{m}) &= \boldsymbol{m}^T \boldsymbol{F}^T \boldsymbol{Q} \boldsymbol{F} \boldsymbol{m} - \boldsymbol{m}^T \boldsymbol{Q} \boldsymbol{m} \\ &= \boldsymbol{m}^T \left(\boldsymbol{F}^T \boldsymbol{Q} \boldsymbol{F} - \boldsymbol{Q} \right) \boldsymbol{m} \\ &= -\boldsymbol{m}^T \boldsymbol{P} \boldsymbol{m}. \end{aligned} \tag{3.66}$$

Contraction Theory The stability of a discrete-time system can also be studied with the contraction Theory [12] using the notion of contraction mapping given by Definition 3.12

Definition 3.12. (Contraction mapping) Given a metric space $(\mathcal{X}, d)$, a mapping $\boldsymbol{h} : \mathcal{X} \to \mathcal{X}$ is a contraction if it is K-Lipschitz with a constant $K < 1$. In this case, K is called the contraction factor of $\boldsymbol{h}$.

Theorem 3.9. *(Banach Contraction Theorem) Consider a metric space $\left(\mathcal{X}, d_{\|\cdot\|}\right)$ with $\mathcal{X} \subseteq \mathbb{R}^n$ and consider the mapping $\boldsymbol{h} : \mathcal{X} \to \mathcal{X}$. If $\boldsymbol{h}$ is a contraction with a contraction factor K, then*

1) $\boldsymbol{h}$ has a unique fixed point $\boldsymbol{m}^*$ in $\mathcal{X}$.

2) the sequence $\{\boldsymbol{m}_k\}_{k \in \mathbb{N}}$ generated by the iteration $\boldsymbol{m}_{k+1} = \boldsymbol{h}(\boldsymbol{m}_k)$ converges to $\boldsymbol{m}^*$ for all initial condition $\boldsymbol{m}_0 \in \mathcal{X}$.

3) the convergence is geometric, i.e. $\forall k \in \mathbb{N}$, $\|\boldsymbol{m}_k - \boldsymbol{m}^*\| \leq K^k \|\boldsymbol{m}_0 - \boldsymbol{m}^*\|$.

Corollary 3.1. *Consider the discrete-time system given by*

$$\begin{aligned} \boldsymbol{m}_{k+1} &= \boldsymbol{h}(\boldsymbol{m}_k), \\ \boldsymbol{0} &= \boldsymbol{h}(\boldsymbol{0}), \end{aligned} \tag{3.67}$$

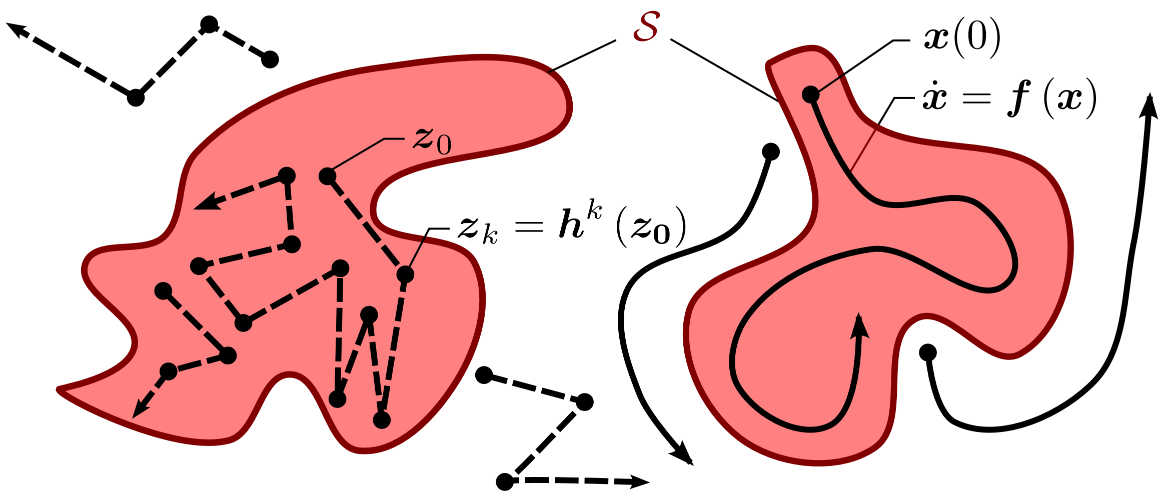
with the mapping $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^n$ and let $\mathcal{X} \subseteq \mathbb{R}^n$. If $\boldsymbol{h}$ is a contraction on $\mathcal{X}$ then the system is exponentially stable on $\mathcal{X}$, i.e. the exist $\alpha > 0$, $\beta > 0$ such that

$$\boldsymbol{m}_0 \in \mathcal{X} \Rightarrow \|\boldsymbol{m}_k\| \leq \alpha \|\boldsymbol{m}_0\| e^{-\beta k}, \forall k \in \mathbb{N}.$$

Proof. Since $\boldsymbol{0}$ is a fixed point of $\boldsymbol{h}$, then from Theorem 3.9, for all $\boldsymbol{m}_0 \in \mathcal{X}$ and all $k \in \mathbb{N}$, one has

$$\begin{aligned} \|\boldsymbol{m}_k\| &\leq K^k \|\boldsymbol{m}_0\| \\ &= e^{\ln(K) \cdot k} \|\boldsymbol{m}_0\|. \end{aligned} \tag{3.68}$$

So the discrete system is exponentially stable on $\mathcal{X}$. □



$\mathcal{S}$
$\boldsymbol{x}(0)$
$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$
$\boldsymbol{z}_0$
$\boldsymbol{z}_k = \boldsymbol{h}^k(\boldsymbol{z}_0)$

Figure 3.16: Positive invariant set $\mathcal{S}$ with respect to the discrete system $\boldsymbol{z}_{k+1} = \boldsymbol{h}(\boldsymbol{z}_k)$ and the continuous system $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$

3.3.3 Positive invariance

From Theorem 3.4, one can prove that the system is locally stable with a linearisation. However, this theorem does not give any information on the size of the region of attraction. When Lyapunov functions are difficult to find, the region of attraction is often described with *positive invariant (PI)* sets [109, 6, 27, 52]. When a set is positive invariant with respect to a system, the trajectory of the system may enter but cannot escape this set. Finding positive invariant sets is a sufficient condition to prove the stability of the system.

In addition, dynamical systems are often subject to a variety of perturbations that affect the systems' stability: the perturbed systems are often only locally stable. In this case, the trajectory of the system will not converge to the equilibrium point, but inside a neighbourhood from which the system cannot escape. For example, consider an autonomous car that maintains a $10\,m$ distance with the car in from of it. If the sensors of the car can measure the actual distance with a precision of $\pm 1\,\mathrm{m}$, then the car won't be able to maintain the distance with a micro precision. Moreover, if the speed of the car ahead is above the maximum speed of the autonomous car, that latter will never be able to catch up. However, one can assume that when both cars respect the speed limitation, the autonomous car will be able to maintain the distance with a meter precision. So one can expect the car to stay in a neighbourhood of the desired distance.

It is thus common to look for positive invariant sets. Some works like [52, 98] propose to compute a precise description of the largest positive invariant set of a function, but with a computation time that is sometimes significant. Some other works propose to find a positive invariant set with little calculations [40].

In this thesis, the background about positive invariance is based on [6]. Positive invariant sets are illustrated by Figure 3.16 and are defined by Definition 3.13 and Definition 3.14.

Definition 3.13. [6, Definition 4.1] Consider the continuous dynamical system

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t)) \tag{3.69}$$

with $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$. The set $\mathcal{S} \subseteq \mathbb{R}^n$ is said to be *positive invariant* (PI) with respect to the system if, for all $\boldsymbol{x}(0) \in \mathcal{S}$, the solution $\boldsymbol{x}(t)$ of system (3.69) verify $\boldsymbol{x}(t) \in \mathcal{S}$ for all $t \geq 0$.

In other words, if the system is initialised in the positive invariant set $\mathcal{S}$, it will remain in it for an infinite amount of time. If the equilibrium point is inside this set, then $\mathcal{S}$ is a stable neighbourhood of the equilibrium point. However, the existence of positive invariant sets can't tell if the system is asymptotically stable: the state may wander in $\mathcal{S}$ without converging towards the equilibrium point.

Positive invariance can also be defined with respect to discrete dynamical systems

Definition 3.14. Consider the discrete dynamical system

$$\boldsymbol{x}_{k+1} = \boldsymbol{h}(\boldsymbol{x}_k) \tag{3.70}$$

with $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^n$. The set $\mathcal{S} \subseteq \mathbb{R}^n$ is said to be *positive invariant* (PI) with respect to the discrete dynamical system (3.70) if for all $\boldsymbol{x}_0 \in \mathcal{S}$, the sequence $(\boldsymbol{x}_k)_{k\in\mathbb{N}}$ verify $\boldsymbol{x}_k \in \mathcal{S}$ for all $k \in \mathbb{N}$.

Positive invariant sets give less information about stability than Lyapunov functions because they only describe the stability of the system at the sets' border. However, for complex systems, these sets are often easier to find than Lyapunov functions.

The notion of positive invariance is related to Lyapunov theory. As presented in Section 3.3.1.4, when a trajectory of the system crosses a level surface of a Lyapunov equation, it enters a set Ω_c and will remain in it, making this set positive invariant. The region of attraction Ω_{att} is also positive invariant.

While it may not be possible to find an analytical expression for positive invariant sets, there are different methods to describe them numerically. For example, one can use Linear Matrix Inequality (LMI) or a sum of squares (SOS) [38]. There are also Interval algorithms that enclose the border of the positive invariant sets [1, 52, 25, 90]. Unfortunately, these algorithmic methods are often exponentially complex with respect to the dimension of the sets. Thus these methods are not effective for a high dimensions problem.

For Hybrid systems, it can be hard to compute positive invariant sets. It is more common to look for *periodic invariant* sets, also called *p-invariant* sets [40]. These sets are periodic invariant meaning that the system's trajectory is periodically in the set. So the system can escape p-invariant sets, but at worst, it returns periodically in it. Of course, a positive invariant set is also p-invariant.

Definition 3.15. The set $\mathcal{A} \subseteq \mathbb{R}^n$ is *periodic positive invariant* (p-invariant) w.r.t. to the continuous dynamical system (3.69) if there is a time $t > 0$ so that for all $\boldsymbol{x}_0 \in \mathcal{A}$, $\phi(t, \boldsymbol{x}_0) \in \mathcal{A}$.

P-invariant sets can also be defined for discrete dynamical systems.

Definition 3.16. The set $\mathcal{A} \subseteq \mathbb{R}^n$ is *periodic positive invariant* (p-invariant) w.r.t. the discrete dynamical system (3.70) if there is a iteration $k \in \mathbb{N}^*$ so that for all $\boldsymbol{x}_0 \in \mathcal{A}$, $\boldsymbol{x}_k = \boldsymbol{f}^k(\boldsymbol{x}_0) \in \mathcal{A}$.

Example 3.5. Figures 3.17 and 3.15 represent some positive invariant sets for the systems of the examples 3.1 and 3.4.

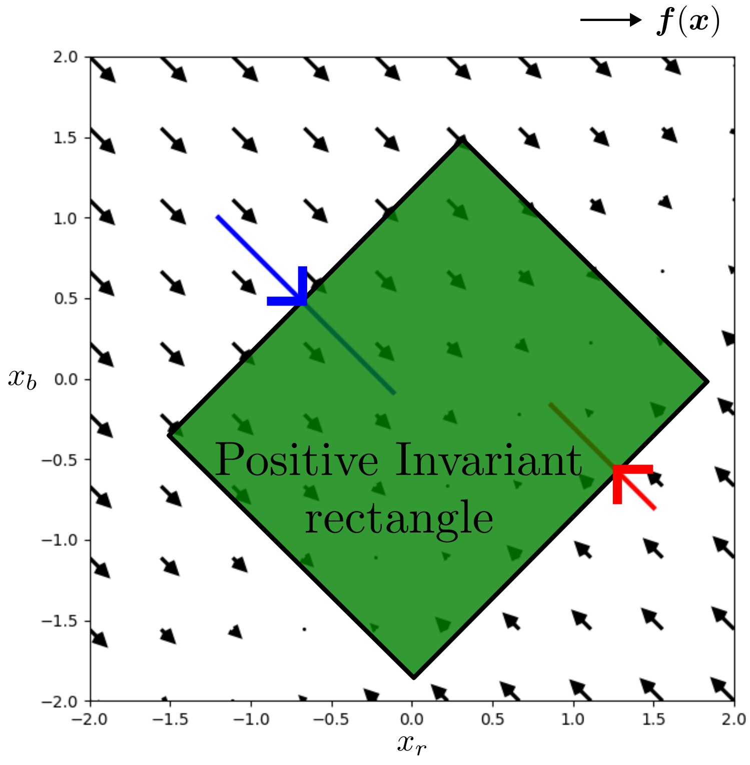


Figure 3.17: For the system of example 3.1, the state cannot escape the positive invariant green rectangle.

3.4 Conclusion

As presented in this chapter, different types of nonlinear systems can describe a group of robots. The stability of these systems is often proved by finding a Lyapunov function: one must propose a candidate function and formally prove that this candidate is a Lyapunov function for the system. Although this method gives strong proof of stability, finding a Lyapunov function is difficult for complex systems, with many variables, a hybrid behaviour, and a lot of non-linearity, as in [111]. Moreover, while the system can be proved locally stable by linearising the system, without a Lyapunov function it is difficult to find positive invariant sets and to describe the region of attraction. When the stability is difficult to study with formal methods, an alternative is to compute positive invariant sets or Lyapunov functions with a guaranteed numerical algorithm. This guarantee means that the solutions of the numerical operations must contain the true mathematical solution. The next chapter will present several numerical tools that have been used to study stability when Lyapunov functions are difficult to find.

Chapter 4

State of Art on numerical stability

4.1 Introduction

This chapter present several numerical tools that can be used for stability analysis. These tools guarantees the result of their computation, so they can be used in a mathematical proof. This guaranty comes form the use of interval algebra presented in Section 4.2. Moreover, the numerical tools developed in this thesis are based one the ellipsoids presented in Section 4.3, a guaranteed numerical method to propagate ellipsoids presented in Section 4.4, and some guaranteed integration methods presented in Section 4.5.

4.2 Interval analysis

The study of dynamical systems is challenging when dealing with non-linearity, noise, external perturbations and uncertainties. In order to overcome these challenges, it is possible to describe linearisation errors and uncertainties as intervals.

With intervals, numerical computations can be guaranteed by computing an rigorous interval of the results. Therefore, these computations can be part of a mathematical proof. For example, interval computations were used to prove the existence of the Lorenz attractor [107]. However, using interval adds pessimism in the computation. Computing with intervals require the use of an interval algebra, which is part of interval analysis.

4.2.1 Background

Interval analysis is a branch of mathematics which was originally developed to solve mathematical problem by enclosing sets of solutions[74, 39, 49]. This enclosure is computed through an arithmetic for intervals, which is often carried out by computers. In interval analysis, a real number x is represented by a pair of numbers a and b such that $a \leq x \leq b$. Instead of using x to solve mathematical problems, operations are carried out on a and b to enclose the round-off errors made by computers. Therefore, interval analysis can be used in rigorous mathematical proof.

Since the beginnings of interval analysis, interval arithmetic has been developed for a variety of operations. In the context of this thesis, interval analysis will be used in the computation of high-dimensional ellipsoids

4.2.2 Interval arithmetic

This section present the key concepts in interval arithmetic as well as the notations, which will be used throughout this thesis. Further details on the concepts are provided in [74, 39].

Definition 4.1. (Interval). An interval written $[x]$ is a connected subset of $\mathbb{R}$. The set of all intervals is denoted $\mathbb{IR}$.

The interval $[x]$ is generally represented by a lower bound written $\underline{x}$ or $\text{lb}\left([x]\right)$ and by an upper bound written $\overline{x}$ or $\text{ub}\left([x]\right)$. These bounds are defined by

$$\underline{x} = \text{lb}\left([x]\right) := \sup\left\{a \in \mathbb{R} \cup \{-\infty, \infty\} \,|\forall x \in [x]\,, a \leq x\right\}, \tag{4.1}$$
$$\overline{x} = \text{ub}\left([x]\right) := \sup\left\{b \in \mathbb{R} \cup \{-\infty, \infty\} \,|\forall x \in [x]\,, x \leq b\right\}. \tag{4.2}$$

With this definition, intervals can be opened if their bound are $-\infty$ or ∞. The width of a closed interval $[x]$ is defined as

$$\text{width}\left([x]\right) := \overline{x} - \underline{x}, \tag{4.3}$$

and the middle of this interval is defined as

$$\text{mid}\left([x]\right) = \frac{\overline{x} + \underline{x}}{2}.$$

Example 4.1. Here are some example of intervals:

- $[-2, 5]$,$[0, \pi]$,$[-\infty, 1]$ are intervals,
- the empty set $\emptyset$ is considered as an interval by convention,
- $[1, 1] = \{1\} \neq \emptyset$ is a degenerate interval,
- the width of $[-2, 5]$ is $\text{width}\left([-2, 5]\right) = 7$,
- the middle of $[-2, 5]$ is $\text{mid}\left([-2, 5]\right) = \frac{3}{2}$.

4.2.2.1 Operations on interval

Common set operations can be performed on intervals. Let $[x]$ and $[y]$ be two intervals of $\mathbb{R}$. Their intersection is defined by

$$[x] \cap [y] := \left\{z \in \mathbb{R} | z \in [x]\,, z \in [y]\right\}, \tag{4.4}$$

and can be computed by

$$[x] \cap [y] = \begin{cases} \emptyset & \text{if } \overline{x} < \underline{y}, \\ \left[\max\left(\underline{x}, \underline{y}\right), \min\left(\overline{x}, \overline{y}\right)\right] & \text{else.} \end{cases} \tag{4.5}$$

While their intersection is always an interval, their union may not. Thus, to make intervals closed with respect to union, the *interval hull* can be used instead. The interval hull is the smallest interval enclosing the union. It is is defined by

$$[x] \sqcup [y] = [[x] \cup [y]], \tag{4.6}$$

and can be computed by

$$[x] \sqcup [y] := \left[\min\left(\underline{x}, \underline{y}\right), \max\left(\overline{x}, \overline{y}\right)\right]. \tag{4.7}$$

Then, usual arithmetic operators can be defined over $\mathbb{IR}$. Let $\diamond \in \{+, -, ., /\}$, then

$$[x] \diamond [y] := \{x \diamond y, x \in [x], y \in [y]\}. \tag{4.8}$$

As for the intersection and the interval hull, these operations can be implemented using interval bounds but special case must be taken regarding multiplication and division or special operator like sin, cos, exp..., see [39].

Example 4.2. Here are some example of operations on intervals

- $[-1, 2] \cap [1, 5] = [1, 2]$
- $[-1, 0] \cap [1, 5] = \emptyset$
- $[-1, 0] \sqcup [1, 5] = [-1, 5]$
- $[-1, 0] + [1, 5] = [0, 5]$
- $[-1, 0] - [1, 5] = [-6, -1]$
- $[-1, 0] \cdot [1, 5] = [-5, 0]$
- $[7, 9] / [2, 5] = \left[\frac{7}{5}, \frac{9}{2}\right]$
- $[1, 5] / [-1, 0] = [-\infty, -1]$

Note that some operation are not intuitive and so must be processed carefully.

4.2.2.2 Interval vectors and matrices

To study multi-variable problems, intervals can be stacked to form interval vectors also called *boxes.*

Definition 4.2. (Box). An *box* written $[\boldsymbol{x}]$ is a connected subset of $\mathbb{R}^n$ defined by a Cartesian product of closed intervals such that

$$[\boldsymbol{x}] := [x_1] \times [x_2] \times \ldots \times [x_n] \tag{4.9}$$

where the interval $[x_i]$ is the projection of $[\boldsymbol{x}]$ into the i^{th} axis. The set of all boxes of $\mathbb{R}^n$ is denoted $\mathbb{IR}^n$.

The width of an closed box $[\boldsymbol{x}]$ is defined as

$$\operatorname{width}([\boldsymbol{x}]) := \max_{1\leq i\leq n} \operatorname{width}([x_i]), \tag{4.10}$$

and the middle of this box is a vector defined as

$$\operatorname{mid}([\boldsymbol{x}]) := [\operatorname{mid}(x_i)]_{1\leq i\leq n}. \tag{4.11}$$

Note that a vector is a degenerate box. Then, boxes can be concatenated to form *interval matrices.*

Definition 4.3. (Interval matrix). An $(m \times n)$-dimensional interval matrix is a subset of $\mathbb{R}^{m\times n}$ that can be defined as the Cartesian product of mn closed intervals. The interval matrix $[\boldsymbol{A}]$ can be written

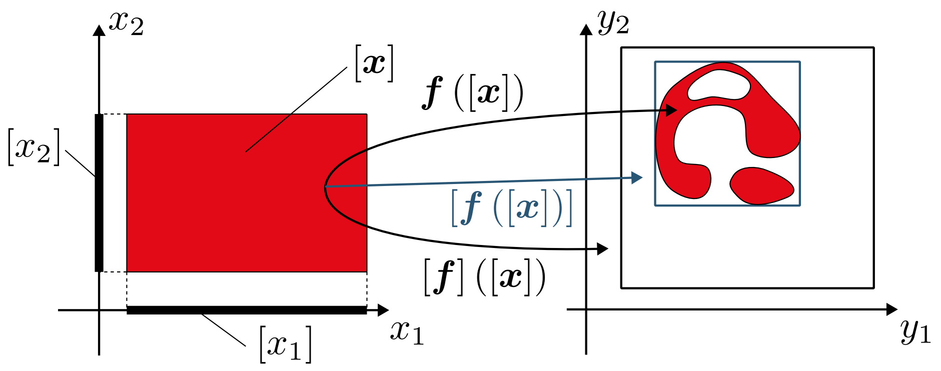
$$\begin{aligned}[\boldsymbol{A}] &:= \begin{pmatrix} [a_{11}] & \cdots & [a_{1n}] \\ \vdots & & \vdots \\ [a_{m1}] & \cdots & [a_{mn}] \end{pmatrix}, \\ &= [a_{11}] \times [a_{12}] \times \ldots \times [a_{mn}], \end{aligned}$$

where $[a_{ij}]$ is the projection of $[A]$ into the (i, j)th axis.

The middle of a closed interval matrix $[\boldsymbol{A}]$ is defined as

$$\operatorname{mid}([\boldsymbol{A}]) := \begin{pmatrix} \operatorname{mid}([a_{11}]) & \cdots & \operatorname{mid}([a_{1n}]) \\ \vdots & & \vdots \\ \operatorname{mid}([a_{m1}]) & \cdots & \operatorname{mid}([a_{mn}]) \end{pmatrix}. \tag{4.12}$$

Moreover, the operations of Section 4.2.2.1 can be applied component-wise to boxes and interval matrices.



x_2
$[\boldsymbol{x}]$
$\boldsymbol{f}([\boldsymbol{x}])$
$[x_2]$
$[\boldsymbol{f}([\boldsymbol{x}])]$
$[\boldsymbol{f}]([\boldsymbol{x}])$
$[x_1]$
x_1
y_2
y_1

Figure 4.1: 2-dimensional box, inclusion function and minimal inclusion function

4.2.2.3 Inclusion functions

In the same way as classical operators, real function can also be extended to intervals.

Definition 4.4. (Inclusion function)[39, Section 2.4]. Consider the function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$. The interval function $[\boldsymbol{f}] : \mathbb{IR}^n \to \mathbb{IR}^m$ is an *inclusion function* for $\boldsymbol{f}$ if

$$\forall [\boldsymbol{x}] \in \mathbb{IR}^n, \boldsymbol{f}([\boldsymbol{x}]) \subset [\boldsymbol{f}]([\boldsymbol{x}]) \tag{4.13}$$

where $\boldsymbol{f}([\boldsymbol{x}]) = \{\boldsymbol{f}(\boldsymbol{x}) | \boldsymbol{x} \in [\boldsymbol{x}]\}$ is the image of the interval $[\boldsymbol{x}]$ by the function $\boldsymbol{f}$.

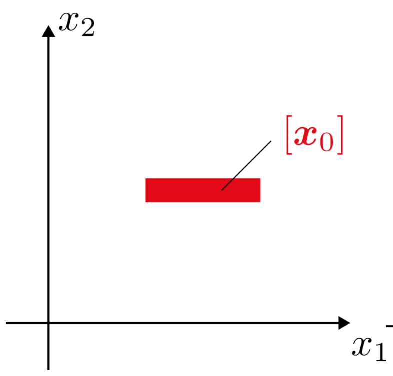
Figure 4.1 illustrates an inclusion function in $\mathbb{R}^2$. While $[\boldsymbol{x}]$ is a box, it is generally not the case for $\boldsymbol{f}([\boldsymbol{x}])$ as in this figure. An inclusion function returns an *enclosure* of the image set $\boldsymbol{f}([\boldsymbol{x}])$ in the form of an interval, a box or interval matrix. The interest is to use operators on inclusion functions as in interval arithmetic. In practice, $[\boldsymbol{f}]([\boldsymbol{x}])$ should be reasonably quickly evaluated but not too large. A large evaluation is commonly called pessimistic. The inclusion function $[\boldsymbol{f}]$ is is called *minimal* if for all $[\boldsymbol{x}] \in \mathbb{IR}^n$, $[\boldsymbol{f}]([\boldsymbol{x}])$ is the smallest box containing the set $\boldsymbol{f}([\boldsymbol{x}])$. It is then denoted by $[\boldsymbol{f}([\boldsymbol{x}])]$.

The main difficulty when using inclusion function is the *wrapping effect*. This phenomenon appears when there are some point in $[\boldsymbol{f}]([\boldsymbol{x}])$ that have an inverse image outside of $[\boldsymbol{x}]$. This wrapping effect is a common cause of pessimism in interval arithmetic. Note that minimal inclusion function introduces the least wrapping effect.

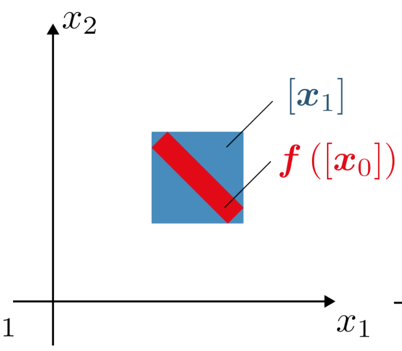
Example 4.3. The Figure 4.2 illustrates the wrapping effect with an iterative rotation of the box around its centre by a $\frac{\pi}{4}$ rad angle. In this example, the function $\boldsymbol{f}$ is defined by

$$\begin{aligned} \boldsymbol{f} :& \mathbb{R}^2 \to \mathbb{R}^2 \\ \boldsymbol{x} \mapsto& \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \cdot \boldsymbol{x}. \end{aligned} \tag{4.14}$$

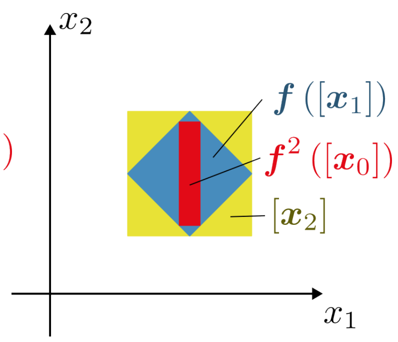
Starting form the box $[\boldsymbol{x}_0]$, at each iteration $\boldsymbol{k}$, the enclosure $[\boldsymbol{x}_k] = [\boldsymbol{f}]([\boldsymbol{x}_{k-1}])$ is computed. At each iteration, $[\boldsymbol{x}_k]$ overestimate $\boldsymbol{f}^k([\boldsymbol{x}_0])$ more and more leading to an enclosure explosion after some steps. This example shows that the composition of inclusion function can lead to bad overestimation.



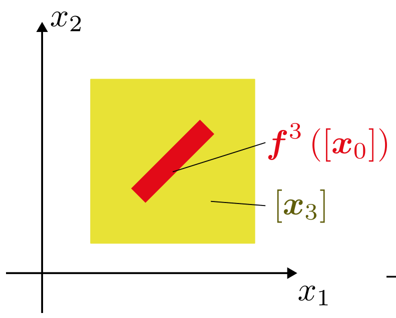
(a) Initial box



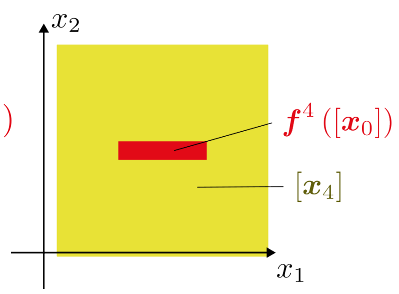
(b) First iteration


x_2
$\boldsymbol{f}([\boldsymbol{x}_1])$
$\boldsymbol{f}^2([\boldsymbol{x}_0])$
$[\boldsymbol{x}_2]$
x_1

(c) Second iteration

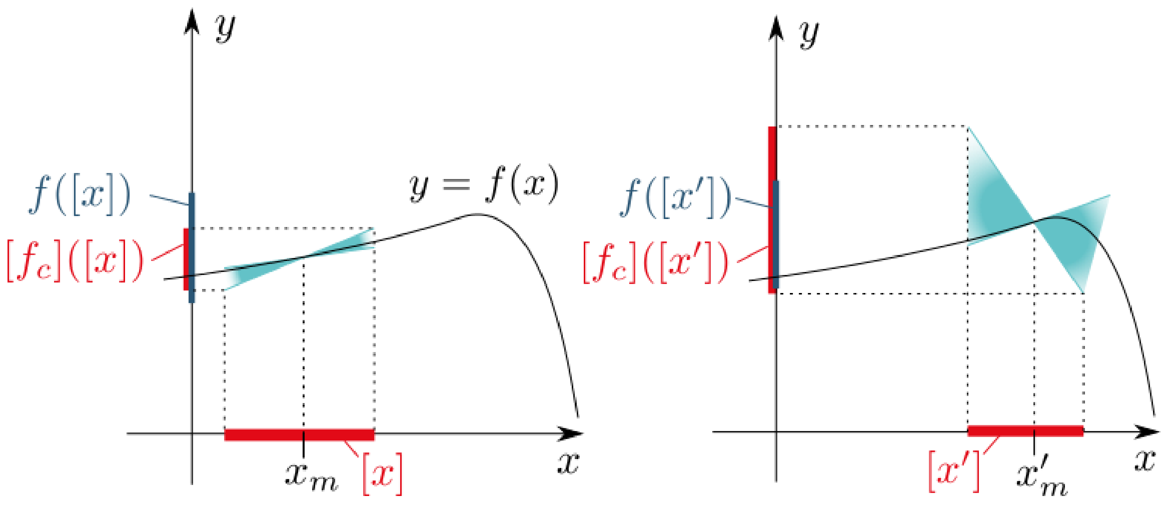


(d) Third iteration



(e) Fourth iteration

Figure 4.2: Illustration of the wrapping effect due to the composition of inclusion functions



(a) Centred form for $[x]$ (b) Centred form for $[x']$

Figure 4.3: Illustration of the centred form

4.2.2.4 Centred form

Some inclusion functions give better approximation of $\boldsymbol{f}([\boldsymbol{x}])$. When $\boldsymbol{f}$ is nonlinear and $[\boldsymbol{x}]$ is small, it is relevant to used a *centred form*, also called a *centred inclusion function.*

Definition 4.5. (Centred form). Consider a differentiable function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$. The *centred form* $[\boldsymbol{f}_c]$ of $\boldsymbol{f}$ is an inclusion function for $\boldsymbol{f}$ defined as

$$\begin{aligned} [\boldsymbol{f}_c] : & \mathbb{IR}^n \to \mathbb{IR}^m \\ & [\boldsymbol{x}] \mapsto \boldsymbol{f}(\boldsymbol{x}_m) + \left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right]([\boldsymbol{x}]) \cdot ([\boldsymbol{x}] - \boldsymbol{x}_m), \end{aligned}$$

with $\boldsymbol{x}_m = \text{mid}([\boldsymbol{x}])$ and where $\left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right]$ is an inclusion function for $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$.

This inclusion function is deduced from the mean-value theorem which implies that

$$\forall \boldsymbol{x} \in [\boldsymbol{x}], \exists \boldsymbol{z} \in [\boldsymbol{x}], \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}_m) + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}(\boldsymbol{z}) \cdot (\boldsymbol{x} - \boldsymbol{x}_m), \tag{4.15}$$

so that $\boldsymbol{f}([\boldsymbol{x}]) \subseteq [\boldsymbol{f}_c]([\boldsymbol{x}])$. The Figure 4.3 illustrate the use of centred form on a 1-dimensional function.

In practice, the centre form results in little wrapping effect when the gradient of the function f is small on $[x]$, which usually happens when $[x]$ is small enough, as in this example. The centred from has a convergence of order 1 in the sense that

$$\frac{\text{width}([\boldsymbol{f}_c]([\boldsymbol{x}]))}{\text{width}(f([\boldsymbol{x}]))} \to 1$$

when width$([\boldsymbol{x}])$ tends to 0. In general, inclusion function don't have this order 1 convergence. However, the centre form often results in more wrapping effect on wide intervals.

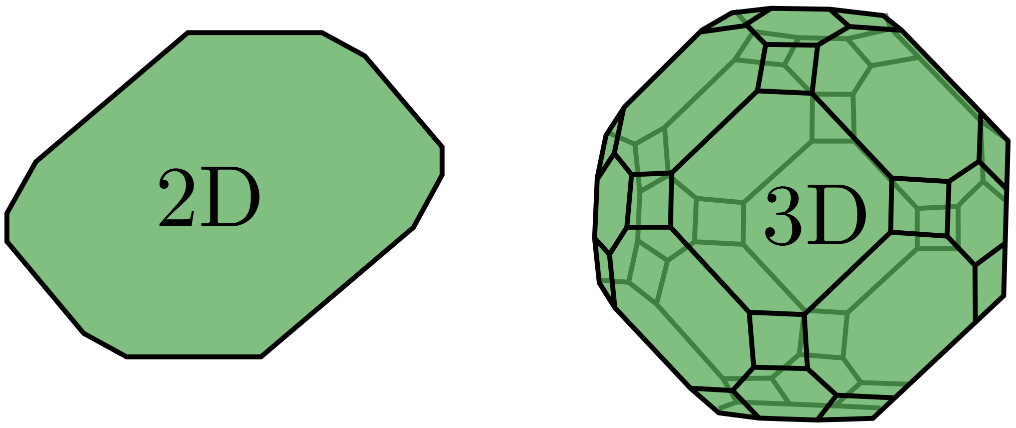


Figure 4.4: Zonotopes are polygons in 2 dimension and polyhedron in 3 dimensions.

4.3 Ellipsoids

As presented in Section 3.3.3, PI set can be an alternative to Lyapunov theory to proof some stability for a dynamical system. There exists some numerical methods to prove the existence of PI sets. These methods are usually designed for sets with simple representations such that

- The result of the numerical method has little pessimism.
- The computational complexity is at worst polynomial.

Thus, in practice, numerical methods are developed based on three types of sets : the *boxes* [9] , the *ellipsoids* and the *zonotopes* [40, 91, 31, 6]. If intervals can be illustrated with a rectangular shape, zonotopes can be described by interval with polygonal shape, like in Figure 4.4.

Boxes give a small computational complexity but a big pessimism. Thus, according to the literature, a good compromise between these two constraints is archived with zonotopes and ellipsoids. According to [1], zonotopes and ellipsoids give similar pessimism and computational complexity with linear systems. Zonotopes can be less pessimistic than the ellipsoids by increasing the number of sides of the zonotopes, at the cost of more computations. Note that a fusion of ellipsoid and zonotopes has been proposed by [47].

Ellipsoids have been used in a variety of approaches to study nonlinear systems. They can be used for reachability analysis [48], stability analysis [58, 92] or state estimation [1, 108]. For linear systems, the library *Ellipsoidal Toolbox* (Matlab) [50] can be used for various ellipsoidal applications. However there is a lack of library for nonlinear systems.

In this thesis, all methods are designed with ellipsoids. This choice was made because ellipsoids are related to Lyapunov function, so perfectly fit for stability analysis. Thus, this section introduces the main mathematical tool of this thesis, the Ellipsoids, which will be used in numerical methods.

4.3.1 Non-degenerate ellipsoids

Ellipsoids are introduced in [87, 95, 6] and are defined by symmetric positive definite matrices. Let S_n^+ be the set of real symmetric positive definite matrix. For a matrix $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$, $\boldsymbol{Q} \succ 0$ means $\boldsymbol{Q}$ is positive definite and $\boldsymbol{Q} \succeq 0$ means $\boldsymbol{Q}$ is semi-definite positive. For each matrix $\boldsymbol{Q} \in \mathcal{S}_n^+$, there exists a unique square root matrix $\boldsymbol{P} \in \mathcal{S}_n^+$ such that $\boldsymbol{Q} = \boldsymbol{P}^2$. One can write $\boldsymbol{P} = \boldsymbol{Q}^{\frac{1}{2}}$. Moreover, for all $\boldsymbol{Q} \in \mathcal{S}_n^+$ the $\boldsymbol{Q}$-norm is defined by

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \|\boldsymbol{x}\|_{\boldsymbol{Q}} := \sqrt{\boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x}}. \tag{4.16}$$

This norm is associated with the scalar product

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\boldsymbol{Q}} := \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{y}. \tag{4.17}$$

Definition 4.6. A non-degenerate ellipsoid is a subset of $\mathbb{R}^n$ described by a unique midpoint $\boldsymbol{\mu} \in \mathbb{R}^n$, a unique shape matrix $\boldsymbol{\Gamma} \in S_n^+$ and the quadratic form

$$\mathcal{E}\left(\boldsymbol{\mu}, \boldsymbol{\Gamma}\right) := \left\{\boldsymbol{x} \in \mathbb{R}^n \middle| \, \|\boldsymbol{x} - \boldsymbol{\mu}\|_{\boldsymbol{\Gamma}^{-2}} \leq 1\right\}. \tag{4.18}$$

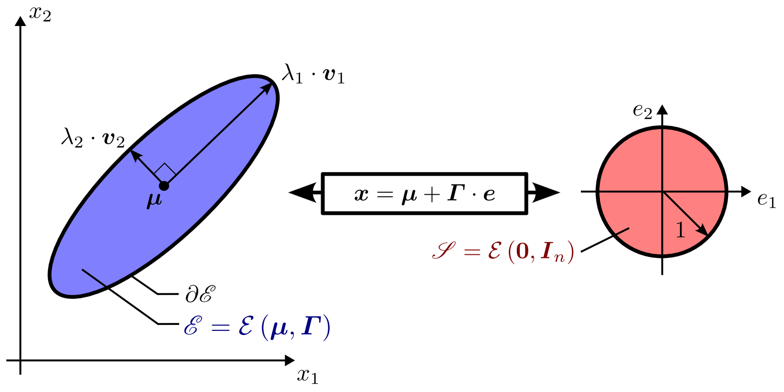

x_2
$\lambda_1 \cdot \boldsymbol{v}_1$
$\lambda_2 \cdot \boldsymbol{v}_2$
$\boldsymbol{\mu}$
$\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{\Gamma} \cdot \boldsymbol{e}$
e_2
e_1
1
$\mathscr{S} = \mathcal{E}(\boldsymbol{0}, \boldsymbol{I}_n)$
$\partial\mathscr{E}$
$\mathscr{E} = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma})$
x_1

Figure 4.5: Representation of a 2-dimensional ellipsoid $\mathscr{E}$

As illustrated by Figure 4.5, a 2-dimensional ellipsoid is an ellipse. An ellipsoid can also be written $\mathscr{E}$ with a midpoint $\boldsymbol{\mu}$ and a shape matrix $\boldsymbol{\Gamma}$ such that

$$\begin{aligned} \mathscr{E} &= \mathcal{E}\left(\boldsymbol{\mu}, \boldsymbol{\Gamma}\right), \\ \left(\boldsymbol{\mu}, \boldsymbol{\Gamma}\right) &= \mathcal{E}^{-1}\left(\mathscr{E}\right). \end{aligned} \tag{4.19}$$

When $\boldsymbol{\mu} = 0$, the ellipsoid $\mathscr{E}$ is said to be centred. For the simplicity of the notation, one may write

$$\mathcal{E}\left(\boldsymbol{\Gamma}\right) = \mathcal{E}\left(\boldsymbol{0}, \boldsymbol{\Gamma}\right). \tag{4.20}$$

The unit sphere is thus written $\mathcal{E}\left(\boldsymbol{I}_n\right)$. The border of an ellipsoid $\mathscr{E}$ is written $\partial\mathscr{E}$. For all $\boldsymbol{\Gamma} \in S_n^+$, all non-null real scalar $\alpha \in \mathbb{R}^*$, and all invertible matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, one has

$$\alpha \cdot \mathcal{E}\left(\boldsymbol{\Gamma}\right) = \mathcal{E}\left(\alpha \cdot \boldsymbol{\Gamma}\right), \tag{4.21}$$

$$\boldsymbol{A} \cdot \mathcal{E}\left(\boldsymbol{\Gamma}\right) = \mathcal{E}\left(\left(\boldsymbol{A}\boldsymbol{\Gamma}^2\boldsymbol{A}^T\right)^{\frac{1}{2}}\right). \tag{4.22}$$

Ellipsoids can also be described as an affine transformation of the unit sphere, as presented in Theorem 4.1 and illustrated in Figure 4.5. This affine description will later be used to define degenerate ellipsoids.

Theorem 4.1. *Let $\mathscr{E}$ be a non-degenerate ellipsoid of $\mathbb{R}^n$ with the midpoint $\boldsymbol{\mu} \in \mathbb{R}^n$ and the shape matrix $\boldsymbol{\Gamma} \in S_n^+$, and let $\mathcal{E}\left(\boldsymbol{I}_n\right)$ be the unit sphere. The ellipsoid $\mathscr{E}$ is equal to*

$$\mathscr{E} = \boldsymbol{\mu} + \boldsymbol{\Gamma} \cdot \mathcal{E}\left(\boldsymbol{I}_n\right). \tag{4.23}$$

Figure 4.5 also illustrates the semi-axis of the ellipsoid which are described by the eigenvalues and the eigenvectors of $\boldsymbol{\Gamma}$. The unitary eigenvectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n\}$ of $\boldsymbol{\Gamma}$ gives the direction of the semi-axis. The length of the semi-axis is given by the eigenvalues $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ of $\boldsymbol{\Gamma}$. The i^{th} semi-axis of $\mathcal{E}\left(\boldsymbol{\mu}, \boldsymbol{\Gamma}\right)$ is written $\lambda_i \cdot \boldsymbol{v}_i$.

4.3.2 Ellipsoids and Lyapunov equations

As presented in Section 3.3, with $\boldsymbol{P} \in \mathcal{S}_n^+$, a quadratic function

$$V\left(\boldsymbol{x}\right) = \boldsymbol{x}^T\boldsymbol{P}\boldsymbol{x} \tag{4.24}$$

can be a Lyapunov function of a continuous-time or discrete-time dynamical system. In this case, the level surface

$$\Omega_c = \left\{\boldsymbol{x} \in \mathbb{R}^n | V\left(x\right) \leq c\right\}, \tag{4.25}$$

with the level $c > 0$, is an ellipsoid. One has

$$\begin{aligned} \Omega_c &= \mathcal{E}\left(\boldsymbol{\Gamma}\right), \\ \boldsymbol{\Gamma} &= \sqrt{c} \cdot \boldsymbol{P}^{-\frac{1}{2}}. \end{aligned} \tag{4.26}$$

Moreover, this level surface is positive invariant with respect to the system. Therefore, finding a positive invariant ellipsoid for a linear dynamical system consists in solving the corresponding Lyapunov equation, as presented by Theorem 4.2 and Theorem 4.3.

Theorem 4.2. *[6, Section 4.4.2]Consider the linear discrete time-invariant dynamical system*

$$\boldsymbol{x}_{k+1} = \boldsymbol{F} \cdot \boldsymbol{x}_k. \tag{4.27}$$

Let $\boldsymbol{Q} \in \mathcal{S}_n^+$ and let $\boldsymbol{P} \in \mathcal{S}_n^+$ be the solution of the discrete time Lyapunov equation

$$\boldsymbol{F}^T \boldsymbol{P} \boldsymbol{F} + \boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0}, \tag{4.28}$$

Then, for all $\alpha > 0$, the ellipsoid $\alpha \cdot \mathcal{E}\left(\boldsymbol{P}^{-\frac{1}{2}}\right)$ is positive invariant with respect to the system (4.27).

Theorem 4.3. *[6, Section 4.4.1]Consider the linear continuous time-invariant dynamical system*

$$\dot{\boldsymbol{x}} = \boldsymbol{A} \cdot \boldsymbol{x}. \tag{4.29}$$

Let $\boldsymbol{Q} \in \mathcal{S}_n^+$ and let $\boldsymbol{P} \in \mathcal{S}_n^+$ be the solution of the continuous time Lyapunov equation

$$\boldsymbol{A}^T \boldsymbol{P} + \boldsymbol{P} \boldsymbol{A} + \boldsymbol{Q} = \boldsymbol{0}, \tag{4.30}$$

Then, for all $\alpha > 0$, the ellipsoid $\alpha \cdot \mathcal{E}\left(\boldsymbol{P}^{-\frac{1}{2}}\right)$ is positive invariant with respect to the system (4.29).

4.3.3 Inclusion of ellipsoids

For two ellipsoids $\mathscr{E}_1$ and $\mathscr{E}_2$, the inclusion and the strict inclusion are respectively denoted by the symbols $\subseteq$ and $\subset$ such that

$$(\mathscr{E}_1 \subset \mathscr{E}_2) \Leftrightarrow (\mathscr{E}_1 \subseteq \mathscr{E}_2 \text{ and } \mathscr{E}_1 \cap \partial \mathscr{E}_2 = \emptyset) \tag{4.31}$$

If two ellipsoids have the same midpoint, their mutual inclusion can be verified using Theorem 4.4. Inclusion is more complex to verify when ellipsoids have different centres. Note that his case will not be studied in this thesis.

Theorem 4.4. *Let $\mathscr{E}_1$ and $\mathscr{E}_2$ be two ellipsoids of $\mathbb{R}^n$ with the same midpoint $\boldsymbol{\mu} \in \mathbb{R}^n$. Consider the shapes matrices $\boldsymbol{\Gamma}_1 \in S_n^+$ and $\boldsymbol{\Gamma}_2 \in S_n^+$ such that $\mathscr{E}_1 = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma}_1)$ and $\mathscr{E}_2 = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma}_2)$. Then, one has*

$$(\mathscr{E}_1 \subseteq \mathscr{E}_2) \Leftrightarrow \left(\boldsymbol{\Gamma}_1^{-2} - \boldsymbol{\Gamma}_2^{-2} \succeq 0\right), \tag{4.32}$$

$$(\mathscr{E}_1 \subset \mathscr{E}_2) \Leftrightarrow \left(\boldsymbol{\Gamma}_1^{-2} - \boldsymbol{\Gamma}_2^{-2} \succ 0\right). \tag{4.33}$$

Note that since $\boldsymbol{\Gamma}_1$ and $\boldsymbol{\Gamma}_2$ are invertible, one gets

$$\begin{aligned}\left(\boldsymbol{\Gamma}_1^{-2} - \boldsymbol{\Gamma}_2^{-2} \succeq 0\right) &\Leftrightarrow \left(\boldsymbol{\Gamma}_2 \boldsymbol{\Gamma}_1^{-2} \boldsymbol{\Gamma}_2 - \boldsymbol{I}_n \succeq 0\right) \\ &\Leftrightarrow \left(\boldsymbol{I}_n - \boldsymbol{\Gamma}_1 \boldsymbol{\Gamma}_2^{-2} \boldsymbol{\Gamma}_1 \succeq 0\right)\end{aligned} \tag{4.34}$$

4.3.4 Orthogonal projection of ellipsoids on affine space

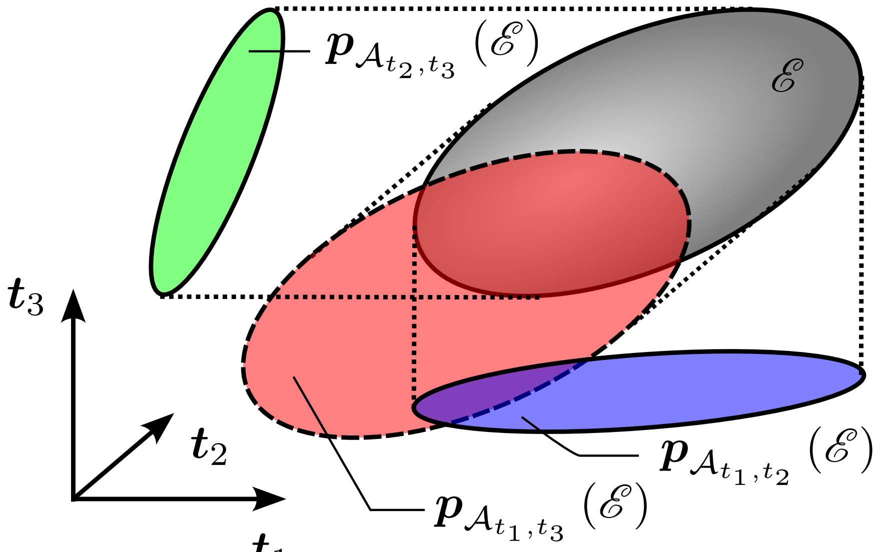
In this thesis, the stability will be analysed with high-dimensional ellipsoids. These ellipsoids will be displayed on 2-dimensional images to give the reader an idea of the ellipsoids' shape. To do so, a list of 2-dimensional orthogonal projection will be displayed. Consider the orthogonal matrix $\boldsymbol{T} = \begin{bmatrix} \boldsymbol{t}_1 & \boldsymbol{t}_2 \end{bmatrix} \in \mathbb{R}^{n\times 2}$ and the plane

$$\mathcal{A}_{t_1,t_2} := \left\{\boldsymbol{x} \in \mathbb{R}^n | \exists \boldsymbol{t} \in \mathbb{R}^2, \boldsymbol{x} = \boldsymbol{T}\cdot\boldsymbol{t}\right\}, \tag{4.35}$$

As presented in [87, Section 13], The projection of an ellipsoid $\mathscr{E}$ of $\mathbb{R}^n$ onto the affine space $\mathcal{A}_{i,j}$ result in the ellipsoid

$$\begin{aligned} \boldsymbol{p}_{\mathcal{A}_{t_1,t_2}}(\mathscr{E}) &= \left(\boldsymbol{T}\boldsymbol{T}^T\right)\cdot\mathscr{E}, \\ &= \mathcal{E}\left(\boldsymbol{T}\boldsymbol{T}^T\boldsymbol{\mu}, \left(\boldsymbol{T}^T\boldsymbol{T}\boldsymbol{\Gamma}^2\boldsymbol{T}\boldsymbol{T}^T\right)^{\frac{1}{2}}\right) \end{aligned} \tag{4.36}$$

with $(\boldsymbol{\mu}, \boldsymbol{\Gamma}) = \mathcal{E}^{-1}(\mathscr{E})$ and such that $\boldsymbol{p}_{\mathcal{A}_{t_1,t_2}}(\mathscr{E}) \subset \mathcal{A}$. Then, with a change of variable $\boldsymbol{y} = \boldsymbol{T}^T\cdot\boldsymbol{x}$, one can display the projected ellipsoid $\boldsymbol{p}_{\mathcal{A}_{t_1,t_2}}(\mathscr{E})$ in the frame $(\boldsymbol{0}, \boldsymbol{t}_1, \boldsymbol{t}_2)$, as illustrated by Figure 4.6. In practice, the ellipsoids will be projected in the orthogonal planes of the Cartesian base $(\boldsymbol{0}, \boldsymbol{e}_i, \boldsymbol{e}_j)$.



$\boldsymbol{p}_{\mathcal{A}_{t_2,t_3}}(\mathscr{E})$
$\mathscr{E}$
$\boldsymbol{t}_3$
$\boldsymbol{t}_2$
$\boldsymbol{t}_1$
$\boldsymbol{p}_{\mathcal{A}_{t_1,t_2}}(\mathscr{E})$
$\boldsymbol{p}_{\mathcal{A}_{t_1,t_3}}(\mathscr{E})$

Figure 4.6: A 3-dimensional ellipsoid can be displayed with three 2-dimensional orthogonal projections with three orthogonal vectors $\boldsymbol{t}_1$, $\boldsymbol{t}_2$ and $\boldsymbol{t}_3$.

4.4 Guaranteed Propagation of ellipsoids

Several methods developed in this thesis will use the propagation of ellipsoids via non-linear function. These approaches follow previous works such as [70, 86, 51] which studied linear or linearised system. In these studies, the propagation of ellipsoids follow an affine arithmetic where stability problems are often equivalent to LMI constraints.

Given an initial ellipsoidal set $\mathscr{E} \subset \mathbb{R}^n$ and the differential nonlinear mapping $\boldsymbol{h} : \mathbb{R}^n \rightarrow \mathbb{R}^n$, the image set

$$\boldsymbol{h}(\mathscr{E}) = \{\boldsymbol{y} \in \mathbb{R}^n | \exists \boldsymbol{x} \in \mathscr{E}, \boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x})\}, \tag{4.37}$$

must be evaluated. As illustrated by Figure 4.7, the result of this propagation will likely be anything but an ellipsoid and will have no analytical expression. It may even become non-convex. It is possible to compute a precise description of $\boldsymbol{h}(\mathscr{E})$ with splitting and sub-paving procedures [46, 52] but these methods have an exponential complexity.

To have polynomial complexity, one can afford to approximate $\boldsymbol{h}(\mathscr{E})$ with an outer enclosure with a simple description, like an ellipsoid. One can therefore look for an ellipsoid $\mathscr{E}_{\text{out}} \subset \mathbb{R}^n$ such that

$$\boldsymbol{h}(\mathscr{E}) \subseteq \mathscr{E}_{\text{out}}, \tag{4.38}$$

Moreover, $\mathscr{E}_{\text{out}}$ should be computed as small as possible with reasonable computation time.

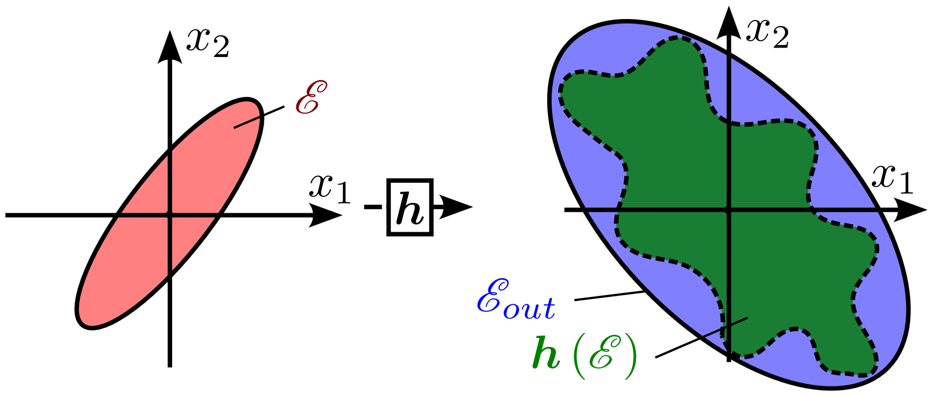


Figure 4.7: Propagation of a 2-dimensional non-degenerate ellipsoid via a nonlinear mapping with the image of the propagation $\boldsymbol{h}(\mathscr{E})$ and an outer enclosure of this image $\mathscr{E}_{\text{out}}$.

4.4.1 Propagation method

The outer ellipsoidal enclosure $\mathscr{E}_{\text{out}}$ can be computed using the method introduced in [95]. The polynomial complexity of this method allows the study of high dimensional mappings. This method, based on the centred form of $\boldsymbol{h}$, keeps the order 1 of convergence and can thus give a good enclosure when $\mathscr{E}$ is small. This method can also study mapping whose analytical expression is unknown, provided that the Jacobian matrix of the mapping is enclosed in a known box. The main assumption of this method is that the Jacobian matrix in invertible. The Theorem 4.5 behind this method is illustrated by Figure 4.8.

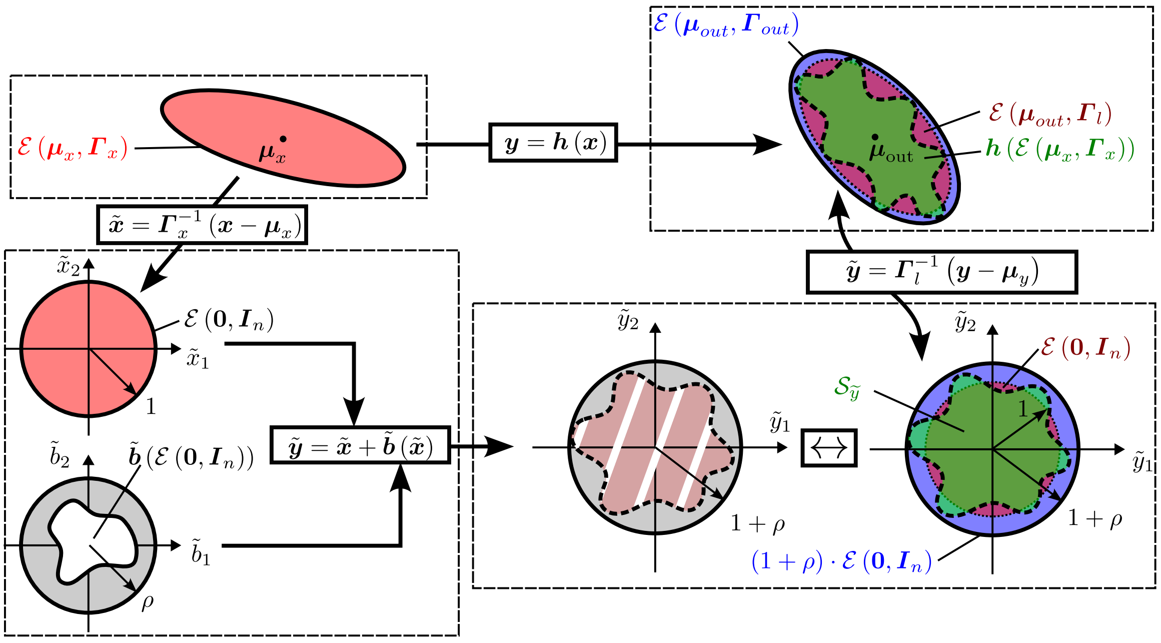

$\mathcal{E}(\boldsymbol{\mu}_x, \boldsymbol{\Gamma}_x)$
$\boldsymbol{\mu}_x$
$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x})$
$\mathcal{E}(\boldsymbol{\mu}_{out}, \boldsymbol{\Gamma}_{out})$
$\mathcal{E}(\boldsymbol{\mu}_{out}, \boldsymbol{\Gamma}_l)$
$\boldsymbol{\mu}_{\text{out}}$
$\boldsymbol{h}(\mathcal{E}(\boldsymbol{\mu}_x, \boldsymbol{\Gamma}_x))$
$\tilde{\boldsymbol{x}} = \boldsymbol{\Gamma}_x^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_x)$
$\tilde{\boldsymbol{y}} = \boldsymbol{\Gamma}_l^{-1}(\boldsymbol{y} - \boldsymbol{\mu}_y)$
$\tilde{x}_2$
$\tilde{x}_1$
$\mathcal{E}(\boldsymbol{0}, \boldsymbol{I}_n)$
1
$\tilde{b}_2$
$\tilde{b}_1$
$\tilde{\boldsymbol{b}}(\mathcal{E}(\boldsymbol{0}, \boldsymbol{I}_n))$
ρ
$\tilde{\boldsymbol{y}} = \tilde{\boldsymbol{x}} + \tilde{\boldsymbol{b}}(\tilde{\boldsymbol{x}})$
$\tilde{y}_2$
$\tilde{y}_1$
$1+\rho$
$\leftrightarrow$
$\mathcal{S}_{\tilde{y}}$
$\tilde{y}_2$
$\mathcal{E}(\boldsymbol{0}, \boldsymbol{I}_n)$
1
$\tilde{y}_1$
$1+\rho$
$(1+\rho) \cdot \mathcal{E}(\boldsymbol{0}, \boldsymbol{I}_n)$

Figure 4.8: Illustration of Theorem 4.5

Theorem 4.5. *[95, Theorem 1] Let $\mathscr{E} = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma})$ be an non-degenerate ellipsoid of $\mathbb{R}^n$ with $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\boldsymbol{\Gamma} \in \mathcal{S}_n^+$. Let $\boldsymbol{h}$ be a $\mathcal{C}^1$ mapping over $\mathbb{R}^n$ such that its Jacobian matrix at the origin,*

$$\boldsymbol{J} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\boldsymbol{\mu}), \tag{4.39}$$

is invertible. Consider the point

$$\boldsymbol{\mu}_{out} = \boldsymbol{h}(\boldsymbol{\mu}), \tag{4.40}$$

and the matrix

$$\boldsymbol{\Gamma}_{out} = (1+\rho)\left(\boldsymbol{J}\boldsymbol{\Gamma}^2\boldsymbol{J}^T\right)^{\frac{1}{2}},$$

where the inflation gain ρ can expressed as

$$\rho = \min\left\{\rho \in \mathbb{R}^+ | \forall \widetilde{\boldsymbol{x}} \in \mathcal{E}(\boldsymbol{I}_n), \left\|\widetilde{\boldsymbol{b}}(\widetilde{\boldsymbol{x}})\right\|_2 \leq \rho\right\}, \tag{4.41}$$

with the function

$$\begin{aligned} \widetilde{\boldsymbol{b}} :& \mathbb{R}^n \mapsto \mathbb{R}^n, \\ & \widetilde{\boldsymbol{x}} \to \boldsymbol{\Gamma}^{-1} \cdot \boldsymbol{J}^{-1} \cdot (\boldsymbol{h}(\boldsymbol{\Gamma} \cdot \widetilde{\boldsymbol{x}} + \boldsymbol{\mu}) - \boldsymbol{\mu}_{out}) - \widetilde{\boldsymbol{x}}. \end{aligned} \tag{4.42}$$

Then, the ellipsoid $\mathscr{E}_{out} = \mathcal{E}(\boldsymbol{\mu}_{out}, \boldsymbol{\Gamma}_{out})$ is an outer enclosure of $\boldsymbol{h}(\mathscr{E})$.

To show this theorem, a first approximation of the set $\boldsymbol{h}(\mathscr{E})$ is made by linearizing $\boldsymbol{h}$ at the point $\boldsymbol{\mu}$, which gives the following linear mapping

$$\begin{aligned} \boldsymbol{h}_l :& \mathbb{R}^n \to \mathbb{R}^n \\ & \boldsymbol{x} \mapsto \boldsymbol{h}(\boldsymbol{\mu}) + \boldsymbol{J} \cdot (\boldsymbol{x} - \boldsymbol{\mu}). \end{aligned} \tag{4.43}$$

The propagation of $\mathscr{E}$ by the linear mapping $\boldsymbol{h}_l$ results in the ellipsoid

$$\begin{aligned}\mathscr{E}_l &= \mathcal{E}\left(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{\Gamma}_l\right) \\ &= \boldsymbol{h}_l\left(\mathscr{E}\right), \end{aligned} \tag{4.44}$$

with,

$$\boldsymbol{\Gamma}_l = \left(\boldsymbol{J}\boldsymbol{\Gamma}^2\boldsymbol{J}^T\right)^{\frac{1}{2}}. \tag{4.45}$$

In practice, when $\mathscr{E}$ is a small ellipsoid, the shape of $\boldsymbol{h}\left(\mathscr{E}\right)$ is closed to $\mathscr{E}_l$. Thus a small inflation of $\mathscr{E}_l$ can result in a good outer-enclosure, as in [51, Theorem 4.1]. Therefore, we will look for an outer enclosure $\mathscr{E}_{\text{out}} = \mathcal{E}\left(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{\Gamma}_{\text{out}}\right)$ with

$$\boldsymbol{\Gamma}_{\text{out}} = (1+\rho)\cdot\boldsymbol{\Gamma}_l, \tag{4.46}$$

while minimising the inflation gain ρ. To compute ρ, the ellipsoids are normalised by the affine transformation $\boldsymbol{y} \mapsto \boldsymbol{\Gamma}^{-1}\cdot\boldsymbol{J}^{-1}\cdot\left(\boldsymbol{y}-\boldsymbol{\mu}_{\text{out}}\right)$ such that $\mathscr{E}_l$ becomes the unit sphere $\mathcal{E}\left(\boldsymbol{I}_n\right)$. With this normalisation, all sets have about a spherical shape. So, ρ is the maximum radius of the set

$$\widetilde{\boldsymbol{b}}\left(\mathcal{E}\left(\boldsymbol{I}_n\right)\right) = \boldsymbol{\Gamma}^{-1}\cdot\boldsymbol{J}^{-1}\cdot\left(\boldsymbol{h}\left(\mathcal{E}\left(\boldsymbol{\mu}_x, \boldsymbol{\Gamma}_x\right)\right) - \boldsymbol{\mu}_{\text{out}}\right). \tag{4.47}$$

4.4.2 Implementation of the propagation method

A numerical implementation of the propagation method is also proposed in [95, Algorithm 1], following Theorem 4.5. Given the function $\boldsymbol{h}$, consider the propagation operator $\mathcal{P}_h$ defined by

$$\mathcal{P}_h\left(\mathscr{E}\right) := \boldsymbol{h}\left(\boldsymbol{\mu}\right) + (1+\rho)\cdot\boldsymbol{J}\cdot\left(\mathscr{E}-\boldsymbol{\mu}\right), \tag{4.48}$$

such that $\mathscr{E}_{\text{out}} = \mathcal{P}_h\left(\mathscr{E}\right)$, with $\boldsymbol{J} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\left(\boldsymbol{\mu}\right)$ and where the computation of ρ is detailed in Algorithm 1. It should be noted that this algorithm is computationally tractable, which is not common in interval algorithms, whose complexity if often exponential. The operator $\mathcal{P}_h$ will simplify the notation in the following chapters.

This algorithm must compute an interval matrix $[\boldsymbol{J}]$ that verifies

$$\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\left(\mathscr{E}\right) \subseteq [\boldsymbol{J}]. \tag{4.49}$$

The method to compute $[\boldsymbol{J}]$ depends on the nature of $\boldsymbol{h}$. In [95, Algorithm 1], where the analytical expression of $\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}$ is known, $[\boldsymbol{J}]$ is computed as

$$[\boldsymbol{J}] = \left[\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right]\left([\mathscr{E}]\right) \tag{4.50}$$

where $[\mathscr{E}]$ is the tightest axis-aligned box that enclose $\mathscr{E}$, given by

$$[\mathscr{E}] = \boldsymbol{\mu} + \operatorname{diag}(\left\|\boldsymbol{\Gamma}_1\right\|_2, \left\|\boldsymbol{\Gamma}_2\right\|_2, ..., \left\|\boldsymbol{\Gamma}_n\right\|_2)\cdot[\boldsymbol{1}]_n \tag{4.51}$$

Algorithm 1 Outer enclosure implmentation

Inputs $\mathscr{E}$

Outputs $\mathcal{P}_h(\mathscr{E})$

```
1: $(\boldsymbol{\mu}, \boldsymbol{\Gamma}) = \mathcal{E}^{-1}(\mathscr{E})$
2: $\boldsymbol{J} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\boldsymbol{\mu})$ // can be an approximation
3: $[\boldsymbol{J}] = \text{EncloseJacobian}(\boldsymbol{h}, \mathscr{E})$ // such that $\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\mathscr{E}) \subseteq [\boldsymbol{J}]$
4: $\left[\widetilde{\boldsymbol{b}}\right] = \left(\boldsymbol{\Gamma}^{-1} \cdot \boldsymbol{J}^{-1} \cdot [\boldsymbol{J}] \cdot \boldsymbol{\Gamma} - \boldsymbol{I}_n\right) \cdot [\mathbf{1}]_n$
5: $\rho = \sup\left\{\left\|\left[\widetilde{\boldsymbol{b}}\right]\right\|\right\}$
6: $\mathcal{P}_h(\mathscr{E}) = \boldsymbol{h}(\boldsymbol{\mu}) + (1+\rho) \cdot \boldsymbol{J} \cdot (\mathscr{E} - \boldsymbol{\mu})$,
```

where $\boldsymbol{\Gamma}_i$ is the i^{th} line of $\boldsymbol{\Gamma}$. To explain the computation of $[\mathscr{E}]$, let $\boldsymbol{x} \in \mathscr{E}$. There exist a point $\widetilde{\boldsymbol{x}} \in \mathcal{E}(\boldsymbol{I}_n)$ such that

$$\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{\Gamma} \cdot \widetilde{\boldsymbol{x}}. \tag{4.52}$$

So the i^{th} component of $\boldsymbol{x}$ is described by

$$x_i = \mu_i + \boldsymbol{\Gamma}_i \cdot \widetilde{\boldsymbol{x}} \tag{4.53}$$

where $\boldsymbol{\Gamma}_i$ is the i^{th} line of $\boldsymbol{\Gamma}$. Knowing that $\|\widetilde{\boldsymbol{x}}\| \leq 1$, the maximum value of the scalar product $\boldsymbol{\Gamma}_i \cdot \widetilde{\boldsymbol{x}}$ is $\|\boldsymbol{\Gamma}_i\|_2$. Thus, one deduces

$$x_i \in \mu_i + \|\boldsymbol{\Gamma}_i\|_2 \cdot [-1, 1]. \tag{4.54}$$

When the analytical expression of $\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}$ is unknown, but $\boldsymbol{h}$ is the flow of a continuous dynamical system, one can compute $[\boldsymbol{J}]$ via a guaranteed integration of the variational equation, as later presented in Section 4.5.2.

Then, using $[\boldsymbol{J}]$, the algorithm compute a box $\left[\widetilde{\boldsymbol{b}}\right]$ that encloses the set $\widetilde{\boldsymbol{b}}(\mathcal{E}(\boldsymbol{I}_n))$, as illustrated by Figure 4.9. The box $\left[\widetilde{\boldsymbol{b}}\right]$ is computed by

$$\left[\widetilde{\boldsymbol{b}}\right] = \left(\boldsymbol{\Gamma}^{-1} \cdot \boldsymbol{J}^{-1} \cdot [\boldsymbol{J}] \cdot \boldsymbol{\Gamma} - \boldsymbol{I}_n\right) \cdot [\mathbf{1}]_n \tag{4.55}$$

To explain the computation of $\left[\widetilde{\boldsymbol{b}}\right]$, let $\widetilde{\boldsymbol{x}} \in \mathcal{E}(\boldsymbol{I}_n)$. Since $\mathcal{E}(\boldsymbol{I}_n)$ is a convex set, from the mean value theorem, there is a point $\widetilde{\boldsymbol{x}}^* \in \mathcal{E}(\boldsymbol{I}_n)$ such that

$$\begin{aligned} \widetilde{\boldsymbol{b}}(\widetilde{\boldsymbol{x}}) &= \widetilde{\boldsymbol{b}}(\mathbf{0}) + \frac{\partial \widetilde{\boldsymbol{b}}}{\partial \widetilde{\boldsymbol{x}}}(\widetilde{\boldsymbol{x}}^*) \cdot (\widetilde{\boldsymbol{x}} - \mathbf{0}) \\ &= \frac{\partial \widetilde{\boldsymbol{b}}}{\partial \widetilde{\boldsymbol{x}}}(\widetilde{\boldsymbol{x}}^*) \cdot \widetilde{\boldsymbol{x}} \end{aligned} \tag{4.56}$$

Then, from the differentiation of Equation (4.42), the Jacobian of $\widetilde{\boldsymbol{b}}$ can be expressed as

$$\begin{aligned}\frac{\partial \widetilde{\boldsymbol{b}}}{\partial \widetilde{\boldsymbol{x}}}(\widetilde{\boldsymbol{x}}) &= \boldsymbol{\Gamma}^{-1} \cdot \boldsymbol{J}^{-1} \cdot \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\boldsymbol{\Gamma} \cdot \widetilde{\boldsymbol{x}} + \boldsymbol{\mu}) \cdot \boldsymbol{\Gamma} - \boldsymbol{I}_n, \\ &= \boldsymbol{\Gamma}^{-1} \cdot \boldsymbol{J}^{-1} \cdot \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\boldsymbol{x}) \cdot \boldsymbol{\Gamma} - \boldsymbol{I}_n. \end{aligned} \tag{4.57}$$

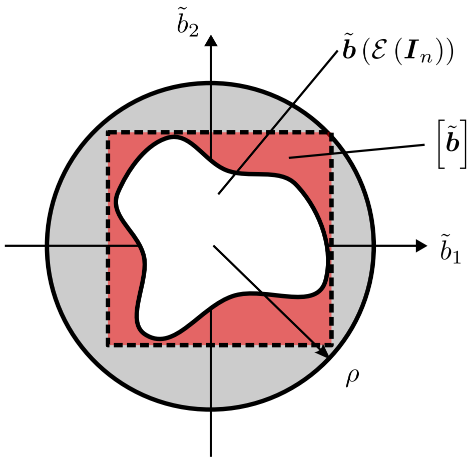
with $\boldsymbol{x} = \boldsymbol{\Gamma} \cdot \widetilde{\boldsymbol{x}} + \boldsymbol{\mu} \in \mathscr{E}$. Therefore, one has

$$\widetilde{\boldsymbol{b}}(\mathcal{E}(\boldsymbol{I}_n)) \subseteq \left(\boldsymbol{\Gamma}^{-1} \cdot \boldsymbol{J}^{-1} \cdot \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\mathscr{E}) \cdot \boldsymbol{\Gamma} - \boldsymbol{I}_n\right) \cdot \mathcal{E}(\boldsymbol{I}_n), \tag{4.58}$$

$$\subseteq \left(\boldsymbol{\Gamma}^{-1} \cdot \boldsymbol{J}^{-1} \cdot [\boldsymbol{J}] \cdot \boldsymbol{\Gamma} - \boldsymbol{I}_n\right) \cdot [\mathbf{1}]_n . \tag{4.59}$$

Finally, the inflation gain ρ is overestimated and computed by

$$\begin{aligned}\rho &= \sup\left\{\left\|\left[\widetilde{\boldsymbol{b}}\right]\right\|\right\} \\ &= \sup\left\{\sqrt{\sum_{i=1}^{n}\left[\widetilde{b}_i\right]^2}\right\}.\end{aligned} \tag{4.60}$$



$\tilde{b}_2$
$\tilde{\boldsymbol{b}}(\mathcal{E}(\boldsymbol{I}_n))$
$[\tilde{\boldsymbol{b}}]$
$\tilde{b}_1$
ρ

Figure 4.9: Illustration of the computation of ρ in Algorithm 1

Example 4.4. This example illustrate the performances of the Algorithm 1

Consider the nonlinear mapping

$$\begin{aligned}\boldsymbol{h} :& \mathbb{R}^2 \to \mathbb{R}^2, \\ &\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \mapsto \begin{bmatrix} x_1 + d_t \cdot x_2 \\ x_2 + d_t \cdot (-\sin(x_1) - 0.5 \cdot x_2^2) \end{bmatrix}\end{aligned} \tag{4.61}$$

with $d_t = 0.4$. This mapping represent a discrete evolution of an nonlinear simple pendulum with damping.

Let us propagate the ellipsoids $\mathscr{E}_1 = \mathcal{E}\left(\boldsymbol{\mu}, \boldsymbol{\Gamma}_1\right)$ and $\mathscr{E}_2 = \mathcal{E}\left(\boldsymbol{\mu}, \boldsymbol{\Gamma}_2\right)$with

$$\begin{aligned}
\boldsymbol{\mu} &= \left[\begin{array}{cc} 2. & 1. \end{array}\right]^T \\
\boldsymbol{\Gamma}_1 &= \left[\begin{array}{cc} 0.4 & 0 \\ 0 & 0.2 \end{array}\right] \\
\boldsymbol{\Gamma}_2 &= 0.1 \cdot \boldsymbol{\Gamma}_1
\end{aligned}$$

using the Algorithm 1. The Figure 4.10 and 4.11 illustrate the result of this propagation. The wrapping effect is clearly visible for $\mathscr{E}_1$. However as $\mathscr{E}_2$ is smaller than $\mathscr{E}_1$ it results in less wrapping effect.

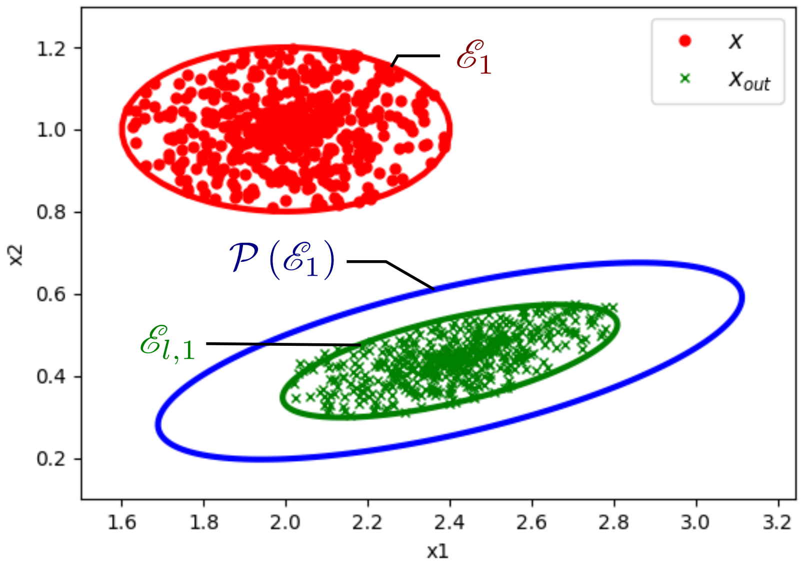


Figure 4.10: Propagation of ellipsoid $\mathscr{E}_1$ by the nonlinear mapping (4.61). The ellipsoid $\mathscr{E}_{l,1} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\left(\boldsymbol{\mu}\right) \cdot \mathscr{E}_1$ results from the linearised mapping. The ellipsoid $\mathcal{P}_h\left(\mathscr{E}_1\right)$ is obtained by inflating $\mathscr{E}_{l,1}$, so that all point $\boldsymbol{x}_{out}$ are inside $\mathcal{P}_h\left(\mathscr{E}_1\right)$. Some 500 points $\boldsymbol{x}$ are sampled in $\mathscr{E}_1$. Their image $\boldsymbol{x}_{\text{out}} = \boldsymbol{h}\left(\boldsymbol{x}\right)$ is inside $\mathcal{P}_h\left(\mathscr{E}_1\right)$.

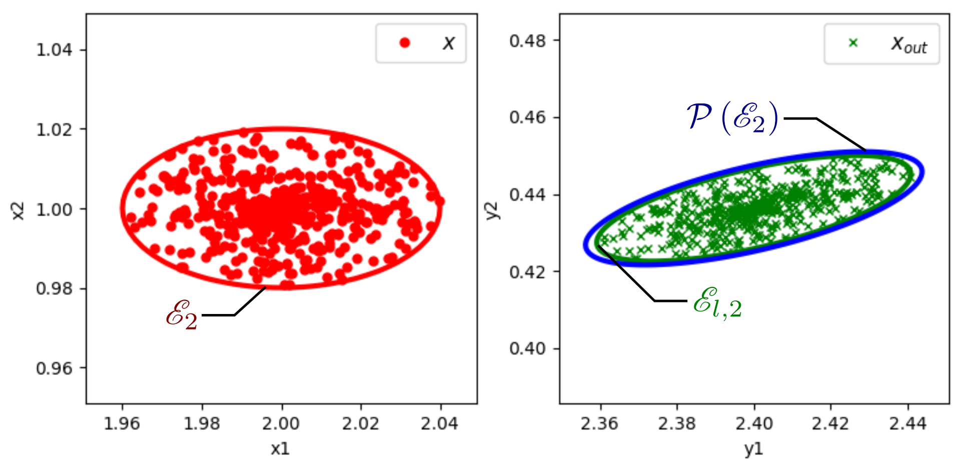


Figure 4.11: Propagation of ellipsoid $\mathscr{E}_2$ by the nonlinear mapping (4.61). The ellipsoid $\mathscr{E}_{l,2} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\boldsymbol{\mu}) \cdot \mathscr{E}_2$ results from the linearised mapping. Then the ellipsoid $\mathcal{P}_h(\mathscr{E}_2)$ is obtained by inflating $\mathscr{E}_{l,2}$. Some 500 points $\boldsymbol{x}$ are Sampled in $\mathscr{E}_2$. Their image $\boldsymbol{x}_{\text{out}} = \boldsymbol{h}(\boldsymbol{x})$ is inside $\mathcal{P}_h(\mathscr{E}_2)$.

In order to apply the method from this section, a challenging task is the computation of the interval matrix of the Jacobian

$$[\boldsymbol{J}] \supseteq \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\mathscr{E}) . \tag{4.62}$$

When the analytical expression of $\boldsymbol{h}$ is known, this computation is trivial. But when $\boldsymbol{h}$ is the flow of a nonlinear continuous-time dynamical equation, then one need to use guaranteed integration algorithms to compute $[\boldsymbol{J}]$.

4.5 Guaranteed integration

For several decades, some mathematical tools have been developed to create algorithms which can perform the integration of differential equations in a guaranteed manner. This means that the result of the algorithm rigorously contains all the reachable state of the predicted system [89]. This Section will present the mathematical tools used for guaranteed integration in this thesis, as well as the associated C++ library which will be used.

The finding of guaranteed enclosure for the integration was already addressed in the 60's by Krückeberg and Moore [49, 72, 73]. They both proposed methods to solve an IVP in a guaranteed way using axis-aligned boxes. But their main problem was the presence of wrapping effect, which led to bloating effect after a few integration steps. Since then, new algorithms have been developed to reduce the wrapping effect to have a sharper enclosure. Moreover, ellipsoids and zonotopes have been considered to replace axis-aligned boxes.

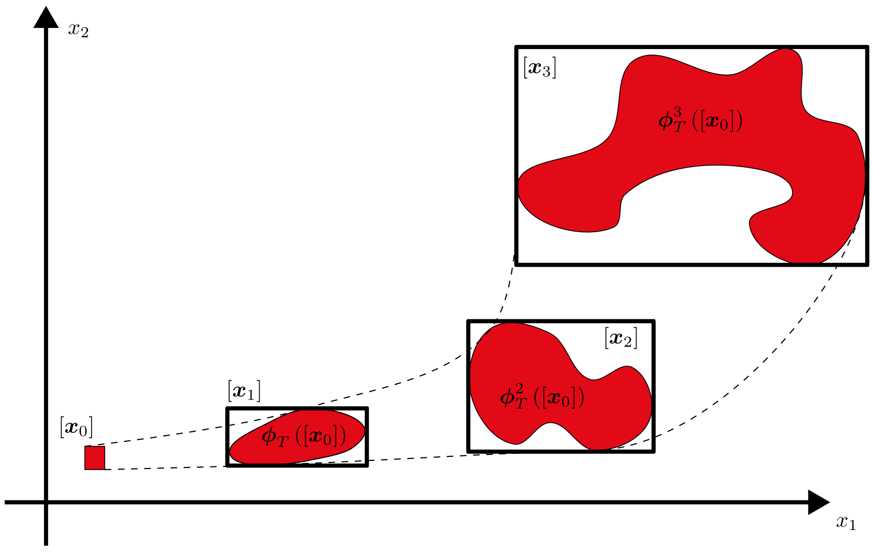

x_2
$[\boldsymbol{x}_3]$
$\boldsymbol{\phi}_T^3([\boldsymbol{x}_0])$
$[\boldsymbol{x}_2]$
$\boldsymbol{\phi}_T^2([\boldsymbol{x}_0])$
$[\boldsymbol{x}_1]$
$\boldsymbol{\phi}_T([\boldsymbol{x}_0])$
$[\boldsymbol{x}_0]$
x_1

Figure 4.12: Rigorous integration of an IVP with the initial condition $[\boldsymbol{x}_0]$

4.5.1 IVP guaranteed solvers

Consider the initial value problem described by

$$\begin{cases} \dot{\boldsymbol{x}}(t) & = \boldsymbol{f}(\boldsymbol{x}(t)), \\ \boldsymbol{x}(0) & \in [\boldsymbol{x}_0], \end{cases} \tag{4.63}$$

as presented in Section 3.2.1. Solving this IVP problem for a time T consist in evaluating the set $\boldsymbol{\phi}_T([\boldsymbol{x}_0])$ with the flow $\boldsymbol{\phi}_T$ of the ODE. Since the analytical expression of $\boldsymbol{\phi}$ is generally unknown, $\boldsymbol{\phi}_T([\boldsymbol{x}_0])$ must be evaluated with a *guaranteed integration* of $\boldsymbol{f}$.

An intuitive solution to integrate the differential inclusion would be to use boxes and an Euler integration scheme with the following equation

$$\begin{aligned} [\boldsymbol{x}](t+dt) &= [\boldsymbol{x}](t) + \int_t^{t+dt} [\boldsymbol{f}]([\boldsymbol{x}](\tau))\,\mathrm{d}\tau, \\ [\boldsymbol{x}](0) &= [\boldsymbol{x}_0], \end{aligned} \tag{4.64}$$

such that $\boldsymbol{\phi}_T([\boldsymbol{x}_0]) \subseteq [\boldsymbol{x}](T)$, as illustrated by Figure 4.12. To integrate $[\boldsymbol{f}]([\boldsymbol{x}](\tau))$, one must find an enclosure $[\boldsymbol{x}_g]$ of all the trajectories starting from $[\boldsymbol{x}](t)$ and evolving for a duration dt. This enclosure is called the *global enclosure* of the solution. Since, there is no direct way to find this global enclosure, iterative methods are often used. Once $[\boldsymbol{x}_g]$ is known, the equation (4.64) becomes

$$[\boldsymbol{x}](t+dt) = [\boldsymbol{x}](t) + [0, dt] \cdot [\boldsymbol{f}]([\boldsymbol{x}_g]). \tag{4.65}$$

This Euler integration method is simple and fast. However, depending on the studied system, its result can be very pessimistic.

Thus, more advanced algorithms have been developed to give more precision in the result. These guaranteed integration algorithms can be split into two groups:

- Those based on the Taylor expansion method [26, 49, 122, 115]
- The ones based on the Hermite-Obreshkov method [78, 112, 67]

The application made in this thesis use only C++ libraries that use the Taylor expansion method.

4.5.2 Using the variational equation to compute the Jacobian of the flow

As presented in Section 3.2.1.1, the Jacobian of a dynamical system's flow, written $\boldsymbol{J}_\phi$, can be computed by integrating the variational equation. The variational equation can be added to the ODE of the system to form the following IVP problem

$$\begin{cases} \dot{\boldsymbol{x}}\left(t\right) & = \boldsymbol{f}\left(\boldsymbol{x}\left(t\right)\right), \\ \dot{\boldsymbol{J}}_\phi\left(t, \boldsymbol{x}\left(0\right)\right) & = \frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{x}_0}\left(\boldsymbol{x}\left(t\right)\right) \cdot \dot{\boldsymbol{J}}_\phi\left(t, \boldsymbol{x}\left(0\right)\right), \\ \boldsymbol{x}\left(0\right) & \in \left[\boldsymbol{x}_0\right], \\ \boldsymbol{J}_\phi\left(0\right) & = \boldsymbol{I}_n. \end{cases} \tag{4.66}$$

Solving this IVP for a time T consist in computing $[\boldsymbol{x}_T]$, $[\boldsymbol{J}_T]$ such that

$$\begin{cases} \boldsymbol{x}\left(T\right) & \in \left[\boldsymbol{x}_T\right], \\ \boldsymbol{J}_\phi\left(T, \boldsymbol{x}\left(0\right)\right) & \in \left[\boldsymbol{J}_T\right]. \end{cases} \tag{4.67}$$

As a result

$$\begin{cases} \boldsymbol{\phi}_T\left(\left[\boldsymbol{x}_0\right]\right) & \in \left[\boldsymbol{x}_T\right], \\ \boldsymbol{J}_\phi\left(T, \left[\boldsymbol{x}_0\right]\right) & \in \left[\boldsymbol{J}_T\right]. \end{cases} \tag{4.68}$$

In particular, if one choose $[\boldsymbol{x}_0] = [\mathscr{E}]$, then the solution $[\boldsymbol{J}_T]$ of the IVP can be used as the Jacobian of the flow

$$\left[\boldsymbol{J}_T\right] = \left[\boldsymbol{J}_\phi\right]\left(T, \left[\mathscr{E}\right]\right), \tag{4.69}$$

that is needed to perform the ellipsoidal propagation method presented in Section 4.4.

4.5.3 Presentation of the CAPD library

In this thesis, the library Computer Assisted Proof in Dynamic groups (CAPD) [41] will be used to perform guaranteed integration. This Library was developed in the 1990's by M. Mrozek to study chaotic dynamics in the Lorenz system [71]. Then Taylor expansion method have been added on an ongoing basis. Their method was initially a C^0 integration based on logarithmic norm. Around 2000, it became a C^1-Lohner algorithm based on Taylor expansion method [122], before becoming a C^r-Lohner algorithm around 2008 [43, 115]. Since then, other chaotic systems have been studied such as the Rössler system or the Michelson system. More recently, this algorithm was enhanced with a Hermite-Obreshkov correction [112]. Here is a link to the documentation of CAPD: https://capd.sourceforge.net/capdDynSys/ .

This library was chosen following the work of a previous thesis [8, 20] that used CAPD in the robotic field. For a detailed presentation of the algorithm in CAPD, see [8]. Note that there exist other libraries which can perform guaranteed integration, such as VNODE-LP [25], DynIbex [15], Valencia-IVP [94] or COSY Infinity [5].

The use of CAPD is here relevant for the following reasons:

- The algorithms have polynomial complexity, which is adapted for high-dimensional systems
- It has an implicit solver for variational equations [112]
- Their method can be used to compute Poincare mapping which could be used to study hybrid systems in future research

Note that this library is designed for boxes and zonotopes but not for ellipsoids. Nevertheless, to solve variational equation, ellipsoid are enclosed in zonotopes, which creates some pessimism.

4.6 Conclusion

With the development of interval analysis, numerical computation can be guaranteed. Thus, these computations can be used in a stability proof, when Lyapunov functions are difficult to find. Moreover, several numerical method, such as the guaranteed propagation of ellipsoid or the guaranteed integration are computationally tractable and can therefore be used on a high-dimensional system such as a group of robots. To reduce the pessimism of the computation, stability should be studied using ellipsoids instead of boxes.

Building on the tools presented in this chapter, the following chapters will present several new guaranteed numerical method to prove stability with ellipsoids. First Chapter 5 will present a method for discrete-time systems. Then Chapter 6 will use the discrete-time method to study continuous-time systems, by linearising them. Chapter 7 will present numerical method with singular mappings and degenerate ellipsoids. Finally, Chapter 8 will use the methods from Chapters 5, 6 and 7 to study synchronous hybrid systems, which better describe a group of robots.

Chapter 5

A guaranteed numerical method to find Ellipsoidal domain of attraction for nonlinear discrete-time system

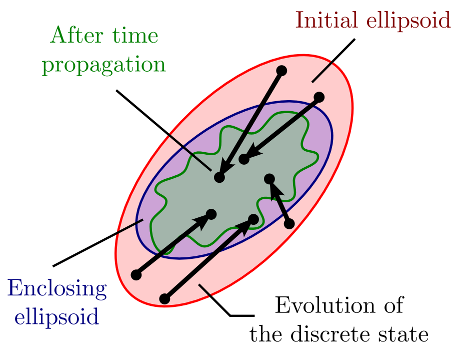
5.1 Introduction

This chapter discusses the stability of discrete-time systems. Although a group of robots is better described by a continuous-time system, which are studies in Chapter6, the numerical methods for continuous systems will be based on the numerical methods for discrete systems.

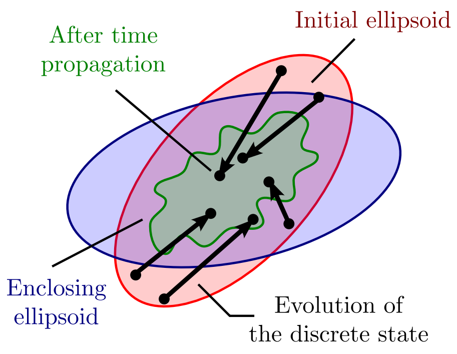
The previous chapter Chapter 3 presented how to use Lyapunov theory to prove the stability of nonlinear discrete-time systems. However, finding a Lyapunov function is difficult for complex high-dimensional systems, such as groups of robots.

Then Chapter 4 presented a numerical method to make guaranteed ellipsoidal propagation. Although many interval analysis algorithms have exponential complexity and are only designed for a small problem dimension, the guaranteed ellipsoidal propagation is computationally tractable. Thus, with this main advantage, this method can be used on high-dimensional problems.

This chapter presents how this numerical tool can be used to prove the stability of discrete-time systems, extending the results of [93]. In [93], the guaranteed ellipsoidal propagation method is used to compute positive invariant ellipsoids for a discrete-time system. This method first solves a discrete Lyapunov equation to find a candidate ellipsoid that is likely positive invariant. From this ellipsoid, an ellipsoidal enclosure of the reachable set is computed using the guaranteed propagation method, as illustrated by Figure 5.1a. If this enclosure is strictly included in the initial ellipsoid, then the ellipsoid is positive invariant with respect to the system. If the inclusion is not verified, as illustrated by Figure 5.1b, the process can be repeated by shrinking the initial ellipsoid, which reduces the pessimism caused by the non-linearity and makes the inclusion more likely to be verified. Of course, if this ellipsoid is not positive invariant, the inclusion cannot be verified, as illustrated by Figure 5.2. To our knowledge, this is the first method that can make a guaranteed computation of a positive invariant ellipsoid for a high-dimensional system.



(a) Good inclusion



(b) Failure of the numerical method

Figure 5.1: Illustration of the numerical method presented in this chapter.

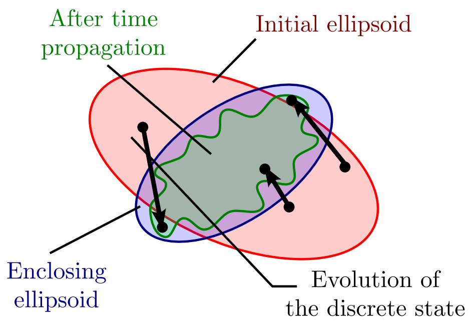


Figure 5.2: Guaranteed propagation of an ellipsoid that is not positive invariant

However, [93] does not provide proof that the system is exponentially stable in the positive invariant ellipsoid. Moreover, the choice of the candidate ellipsoid is not efficient when there is rotation in the system's mapping: the enclosure and the initial ellipsoids have different axes which makes the ellipsoidal inclusion less likely to be verified.

This chapter extends the result of [93] on two points:

1. The guaranteed ellipsoidal propagation can also prove that the discrete-time system is exponentially stable in the positive invariant ellipsoid.

2. Another particular case of the Lyapunov equation is considered to improve the chance of success of the algorithm, by selecting ellipsoids that keep their axis with the propagation.

This chapter is organised as follows. Section 5.2 presents the stability problem considered in this chapter. Section 5.3 presents a key theorem to prove stability using ellipsoidal propagation. Section 5.4 presents a method to find ellipsoids that are likely to be positive invariant. Section 5.5 present the numerical implementation of the stability analysis method described in the previous sections. Finally, Section 5.6 shows an application of the method on a formation control problem with two ROVs.

5.2 Problem definition

In this Chapter, consider a nonlinear discrete-time system, introduced in Section 3.2.2 and described by the following recursive formula

$$\boldsymbol{m}_{k+1} = \boldsymbol{h}\left(\boldsymbol{m}_k\right), \tag{5.1}$$

where $\boldsymbol{m}_k \in \mathbb{R}^n$ is the state vector of the system at the time $t_k = k \cdot T$ with $k \in \mathbb{N}$ and $T > 0$ and where :

- the mapping $\boldsymbol{h}$ is $\mathcal{C}^1$ but its analytical expression is known.
- the system is represented such that the origin $\mathbf{0}$ of $\mathbb{R}^n$ is an equilibrium point, with $\mathbf{0} = \boldsymbol{h}\left(\mathbf{0}\right)$.
- the system is assumed locally exponentially stable, so the Jacobian at the origin

  $$\boldsymbol{J} = \frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}\left(\mathbf{0}\right) \tag{5.2}$$

  is a Schur matrix.

Note that an extension of the stability analysis method to study continuous or hybrid systems by discretising the system will be discussed in Section 5.5.

Since the system is high-dimensional, the computation of the domain of attraction

$$\Omega_{\text{att}} = \left\{\boldsymbol{m}_0 \in \mathbb{R}^n | \lim_{k\to\infty} \boldsymbol{h}^k\left(\boldsymbol{m}_0\right) = \mathbf{0}\right\} \tag{5.3}$$

with a reasonable computation time is challenging. To evaluate the domain of attraction, the problem of this chapter consists in computing a positive invariant ellipsoid $\mathscr{E}$ included in the domain of attraction ($\mathscr{E} \subseteq \Omega_{\text{att}}$) such that the system is locally exponentially stable in $\mathscr{E}$, i.e. there exist $\gamma \in]0,1[$ that verify

$$\boldsymbol{m}_0 \in \mathscr{E} \Rightarrow \left\|\boldsymbol{m}_k\right\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha \left\|\boldsymbol{m}_0\right\|_{\boldsymbol{\Gamma}^{-2}} e^{k\cdot\ln\gamma}, \forall k \in \mathbb{N}, \tag{5.4}$$

with $\boldsymbol{\Gamma} = \mathcal{E}^{-1}\left(\mathscr{E}\right)$. The problem is decomposed into two steps:

- Proving the exponential stability of the system in an ellipsoid $\mathscr{E}$, as presented in Section 5.3.
- Finding an ellipsoid $\mathscr{E}$ that is likely positive invariant, as presented in Section 5.4.

5.3 Proof of the exponential stability

The exponential stability in $\mathscr{E}$ can be proved with Theorem 5.1, illustrated by Figure 5.3, and that relies on Lemma 5.1. The inclusion 5.1 can be numerically verified as presented in Section 5.5.

Lemma 5.1. *Consider the discrete system (5.1) with the propagation operator $\mathcal{P}_h$. For every real $\alpha \in]0,1]$ and every ellipsoid $\mathscr{E}$, one has*

$$\mathcal{P}_h\left(\alpha \cdot \mathscr{E}\right) \subseteq \alpha \cdot \mathcal{P}_h\left(\mathscr{E}\right). \tag{5.5}$$

Proof. Consider the ellipsoid $\mathscr{E}$ and the matrix $\boldsymbol{\Gamma} = \mathcal{E}^{-1}\left(\mathscr{E}\right)$. From Theorem 4.5, one has

$$\mathcal{P}_h\left(\mathscr{E}\right) = \mathcal{E}\left(\boldsymbol{\Gamma}_{out}\right), \tag{5.6}$$

with the matrix

$$\boldsymbol{\Gamma}_{out} = (1+\rho)\left(\boldsymbol{J}^T\boldsymbol{\Gamma}^2\boldsymbol{J}\right)^{\frac{1}{2}},$$

with the inflation gain

$$\rho = \mathrm{ub}\left(\left\|\left(\boldsymbol{\Gamma}^{-1}\cdot\boldsymbol{J}^{-1}\cdot[\boldsymbol{J}]\cdot\boldsymbol{\Gamma} - \boldsymbol{I}_n\right)\cdot[1_n]\right\|\right), \tag{5.7}$$

where the interval matrix $[\boldsymbol{J}]$ verifies

$$\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}\left(\mathscr{E}\right) \subseteq [\boldsymbol{J}]. \tag{5.8}$$

Then let $\alpha \in]0,1]$. One has $\alpha \cdot \mathscr{E} \subset \mathscr{E}$, so

$$\begin{aligned}\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}\left(\alpha\cdot\mathscr{E}\right) &\subseteq \frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}\left(\mathscr{E}\right),\\ &\subseteq [\boldsymbol{J}].\end{aligned} \tag{5.9}$$

Therefore, at least $\mathcal{P}_h\left(\alpha\cdot\mathscr{E}\right)$ can be computed with $[\boldsymbol{J}]$ such that

$$\begin{aligned}\mathcal{P}_h\left(\alpha\cdot\mathscr{E}\right) &= \mathcal{P}_h\left(\mathcal{E}\left(\alpha\boldsymbol{\Gamma}\right)\right),\\ &= \mathcal{E}\left(\boldsymbol{\Gamma}_{out,\alpha}\right).\end{aligned} \tag{5.10}$$

with the matrix

$$\boldsymbol{\Gamma}_{out,\alpha} = (1+\rho_\alpha)\left(\boldsymbol{J}^T\left(\alpha\boldsymbol{\Gamma}\right)^2\boldsymbol{J}\right)^{\frac{1}{2}}, \tag{5.11}$$

where the inflation gain verifies

$$\begin{aligned}\rho_\alpha &\leq \mathrm{ub}\left(\left\|\left(\left(\alpha\boldsymbol{\Gamma}\right)^{-1}\cdot\boldsymbol{J}^{-1}\cdot[\boldsymbol{J}]\cdot\left(\alpha\boldsymbol{\Gamma}\right) - \boldsymbol{I}_n\right)\cdot[1_n]\right\|\right),\\ &= \mathrm{ub}\left(\left\|\left(\boldsymbol{\Gamma}^{-1}\cdot\boldsymbol{J}^{-1}\cdot[\boldsymbol{J}]\cdot\boldsymbol{\Gamma} - \boldsymbol{I}_n\right)\cdot[1_n]\right\|\right)\\ &= \rho.\end{aligned} \tag{5.12}$$

Moreover, from (5.11), one has

$$\boldsymbol{\Gamma}_{out,\alpha} = \frac{\alpha\cdot(1+\rho_\alpha)}{(1+\rho)}\cdot\left((1+\rho)\left(\boldsymbol{J}^T\boldsymbol{\Gamma}^2\boldsymbol{J}\right)^{\frac{1}{2}}\right), \tag{5.13}$$

$$= \alpha\cdot\gamma\cdot\boldsymbol{\Gamma}_{out}, \tag{5.14}$$

with

$$\gamma = \frac{(1+\rho_\alpha)}{(1+\rho)}. \tag{5.15}$$

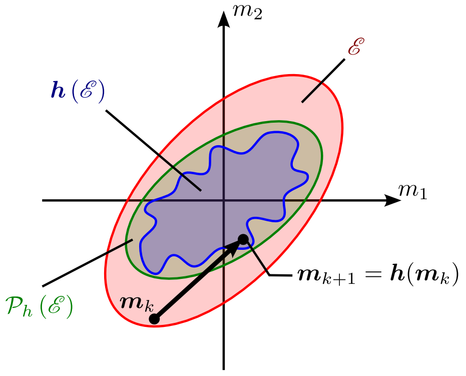
Thus, one obtains

$$\begin{aligned}\mathcal{P}_h(\alpha\cdot\mathscr{E}) &= \mathcal{E}(\boldsymbol{\Gamma}_{out,\alpha}),\\ &= \mathcal{E}(\alpha\cdot\gamma\cdot\boldsymbol{\Gamma}_{out}),\\ &= \alpha\cdot\gamma\cdot\mathcal{E}(\boldsymbol{\Gamma}_{out}),\\ &= \alpha\cdot\gamma\cdot\mathcal{P}_h(\mathscr{E}). \end{aligned} \tag{5.16}$$

Then, from (5.12) one gets $\gamma \in]0,1]$. Therefore

$$\mathcal{P}_h(\alpha\cdot\mathscr{E}) \subseteq \alpha\cdot\mathcal{P}_h(\mathscr{E}). \tag{5.17}$$

□



m_2
m_1
$\mathscr{E}$
$\boldsymbol{h}(\mathscr{E})$
$\mathcal{P}_h(\mathscr{E})$
$\boldsymbol{m}_k$
$\boldsymbol{m}_{k+1} = \boldsymbol{h}(\boldsymbol{m}_k)$

Figure 5.3: Illustration of Theorem 5.1

Theorem 5.1. *Consider the discrete system (5.1) with the propagation operator $\mathcal{P}_h$. If there exists an ellipsoid $\mathscr{E}$ such that*

$$\mathcal{P}_h(\mathscr{E}) \subset \mathscr{E}, \tag{5.18}$$

then $\mathscr{E}$ is positive invariant with respect to the system (5.1) and the system (5.1) is locally exponentially stable on the ellipsoid $\mathscr{E}$.

In Theorem 5.1, the positive invariance is deduced from the fact that $\boldsymbol{h}(\mathscr{E}) \subseteq \mathcal{P}_h(\mathscr{E})$. The exponential stability is deduced from the Lemma 5.1 as following

Proof. Consider an ellipsoid $\mathscr{E}$ such that

$$\mathcal{P}_h(\mathscr{E}) \subset \mathscr{E}, \tag{5.19}$$

and let $\boldsymbol{\Gamma} = \mathcal{E}^{-1}(\mathscr{E})$. So, from (4.31) one has

$$\mathcal{P}_h(\mathscr{E}) \subseteq \mathscr{E} \text{ and } \mathcal{P}_h(\mathscr{E}) \cap \partial\mathscr{E} = \emptyset. \tag{5.20}$$

Moreover, the ellipsoid $\mathcal{P}_h(\mathscr{E})$ is a compact set, so the norm $\|.\|_{\boldsymbol{\Gamma}^{-2}}$ is bounded above $\mathcal{P}_h(\mathscr{E})$ and attains its supremum $\gamma \in \mathbb{R}^+$. By absurd, $\gamma \geq 1$ contradicts (5.20). Moreover, since the mapping $\boldsymbol{h}$ is not singular, one has $\mathcal{P}_h(\mathscr{E}) \neq \{\boldsymbol{0}\}$ and so $\gamma > 0$.

Therefore, one gets $\gamma \in]0, 1[$ and

$$\mathcal{P}_h(\mathscr{E}) \subset \gamma \cdot \mathscr{E}. \tag{5.21}$$

Then, let $\boldsymbol{m} \in \mathscr{E}$ and $\alpha = \|\boldsymbol{m}\|_{\boldsymbol{\Gamma}^{-2}}$. So $\boldsymbol{m} \in \alpha \cdot \mathscr{E}$. Moreover, from Lemma 5.1, one has

$$\mathcal{P}_h(\alpha \cdot \mathscr{E}) \subseteq \alpha \cdot \mathcal{P}_h(\mathscr{E}). \tag{5.22}$$

In addition, from Theorem 4.5, one gets

$$\boldsymbol{h}(\alpha \cdot \mathscr{E}) \subseteq \cdot\mathcal{P}_h(\alpha \cdot \mathscr{E}). \tag{5.23}$$

Thus, from (5.21), (5.22) and (5.23), one deduces

$$\boldsymbol{h}(\alpha \cdot \mathscr{E}) \subset \gamma \cdot \alpha \cdot \mathscr{E}. \tag{5.24}$$

As a consequence

$$\|\boldsymbol{h}(\boldsymbol{m})\|_{\boldsymbol{\Gamma}^{-2}} < \gamma \cdot \alpha = \gamma \cdot \|\boldsymbol{m}_0\|_{\boldsymbol{\Gamma}^{-2}}. \tag{5.25}$$

So $\boldsymbol{h}$ is a pseudocontraction. As a result, from Theorem 3.9, the discrete system is exponentially stable in $\mathscr{E}$. □

5.4 Axis-aligned Lyapunov equation

To verify the requirement of Theorem 5.1, one must find an ellipsoid $\mathscr{E}$ that verifies

$$\mathcal{P}_h(\mathscr{E}) \subset \mathscr{E}, \tag{5.26}$$

In practice, Theorem 5.2 can find an ellipsoid that will likely verify (5.26).

Theorem 5.2. *(Axis-aligned discrete Lyapunov equation) Let $\boldsymbol{J} \in \mathbb{R}^{n\times n}$ be a Schur matrix and let $\boldsymbol{P} \in \mathcal{S}_n^+$ be the unique solution to the discrete Lyapunov equation*

$$\boldsymbol{J}^T\boldsymbol{P}\boldsymbol{J} - \boldsymbol{P} = -\boldsymbol{Q}, \tag{5.27}$$

with $\boldsymbol{Q} = \boldsymbol{J}^T\boldsymbol{J}$. Let $\boldsymbol{\Gamma} = \boldsymbol{P}^{-\frac{1}{2}}$ and $\boldsymbol{\Gamma}_J = \left(\boldsymbol{J}\boldsymbol{\Gamma}^2\boldsymbol{J}^T\right)^{\frac{1}{2}}$. Then $\boldsymbol{\Gamma}$ and $\boldsymbol{\Gamma}_J$ have the same eigenvectors and $\mathcal{E}(\boldsymbol{\Gamma}_J) \subset \mathcal{E}(\boldsymbol{\Gamma})$.

Proof. From (3.65), $\mathcal{E}(\boldsymbol{\Gamma})$ is positive invariant with respect to the linear mapping

$$\boldsymbol{m}_{k+1} = \boldsymbol{J} \cdot \boldsymbol{m}_k. \tag{5.28}$$

Thus

$$\boldsymbol{J} \cdot \mathcal{E}(\boldsymbol{\Gamma}) \subset \mathcal{E}(\boldsymbol{\Gamma}). \tag{5.29}$$

Then, one has

$$\boldsymbol{\Gamma}_J^{-2} = \boldsymbol{J}^{-T}\boldsymbol{\Gamma}^{-2}\boldsymbol{J}^{-1}. \tag{5.30}$$

Thus, from (5.27), one has

$$\begin{aligned} \boldsymbol{J}^T\boldsymbol{P}\boldsymbol{J} - \boldsymbol{P} &= -\boldsymbol{J}^T\boldsymbol{J}, \\ \boldsymbol{P} - \boldsymbol{J}^{-T}\boldsymbol{P}\boldsymbol{J}^{-1} &= -\boldsymbol{I}_n \\ \boldsymbol{\Gamma}^{-2} - \boldsymbol{J}^{-T}\boldsymbol{\Gamma}^{-2}\boldsymbol{J}^{-1} &= -\boldsymbol{I}_n \\ \boldsymbol{\Gamma}^{-2} - \boldsymbol{\Gamma}_J^{-2} &= -\boldsymbol{I}_n \end{aligned} \tag{5.31}$$

Therefore, $\boldsymbol{\Gamma}^{-2}$ and $\boldsymbol{\Gamma}_J^{-2}$ have the same eigenvectors. So $\boldsymbol{\Gamma}^{-2}$ and $\boldsymbol{\Gamma}_J^{-2}$ also have the same eigenvector. □

Example 5.1. For example, consider the linear discrete-time system

$$\begin{aligned} \boldsymbol{m}_{k+1} &= \boldsymbol{m}_k + d_t \cdot \begin{bmatrix} m_{2,k} \\ -m_{1,k} - m_{2,k} \end{bmatrix} \\ &= \boldsymbol{J} \cdot \boldsymbol{m}_k, \end{aligned} \tag{5.32}$$

with

$$\boldsymbol{J} = \begin{bmatrix} 1. & d_t \\ -d_t & 1 - d_t \end{bmatrix}. \tag{5.33}$$

With $d_t = 0.5$, and by writing the solution of the equation (3.65)

$$\boldsymbol{P} = \begin{bmatrix} a & b \\ b & c \end{bmatrix}, \tag{5.34}$$

the equation (3.65) can be rewritten

$$\begin{cases} -b + 0.25c + 1.25 &= 0, \\ 0.5a - 0.75b - 0.25c + 0.25 &= 0, \\ 0.25a + 0.5b - 0.75c + 0.5 &= 0. \end{cases} \tag{5.35}$$

The solution of this system is $\left(a = \frac{59}{13}, b = \frac{28}{13}, c = \frac{47}{13}\right)$. As a result, the matrices of the ellipsoid $\mathcal{E}(\boldsymbol{\Gamma})$ and $\mathcal{E}(\boldsymbol{\Gamma}_J)$ are

$$\begin{aligned} \boldsymbol{\Gamma} &= \boldsymbol{P}^{-\frac{1}{2}} \\ &\simeq \begin{bmatrix} 0.53 & -0.16 \\ -0.16 & 0.60 \end{bmatrix}, \end{aligned} \tag{5.36}$$

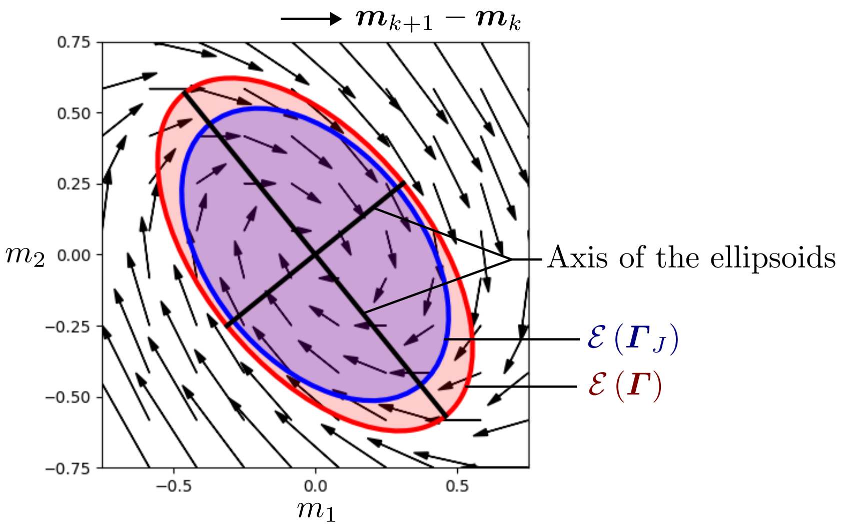


Figure 5.4: Axis aligned ellipsoid for the system (5.32) with $d_t = 0.5$.

and

$$\begin{aligned}\boldsymbol{\Gamma}_J &= \left(\boldsymbol{J}\boldsymbol{\Gamma}^2\boldsymbol{J}^T\right)^{\frac{1}{2}} \\ &\simeq \begin{bmatrix} 0.46 & -0.11 \\ -0.11 & 0.50 \end{bmatrix}.\end{aligned} \tag{5.37}$$

As illustrated by Figure 5.4, these two ellipsoids have the same axis.

5.5 Implementation

Algorithm 2 Computation of the positive invariant ellipsoid $\mathcal{E}\left(\boldsymbol{\Gamma}\right)$ in which the system is exponentially stable

```
Inputs $\boldsymbol{h}$,$\alpha_{\max}$
Outputs res, $\boldsymbol{\Gamma}$
 1: $\boldsymbol{J} = \frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}(\boldsymbol{0})$
 2: $\boldsymbol{P}$ = solve_discrete_Lyapunov($\boldsymbol{J}$)
 3: $\alpha = 0$
 4: res = False
 5: while $\alpha < \alpha_{\max}$ and res = False do
 6:     $\boldsymbol{\Gamma} = 10^{-\alpha} \cdot \boldsymbol{P}^{-\frac{1}{2}}$
 7:     $\boldsymbol{\Gamma}_{out} = \mathcal{E}^{-1}\left(\mathcal{P}_h\left(\mathcal{E}\left(\boldsymbol{\Gamma}\right)\right)\right)$
 8:     res = is_definite_positive$\left(\boldsymbol{\Gamma}_{out}^{-2} - \boldsymbol{\Gamma}^{-2}\right)$
 9:     $\alpha = \alpha + 1$
10: end while
```

The Algorithm 2 is proposed to solve the problem introduced in Section 5.2, using the results from Section 5.3 and Section 5.4. This algorithm is computationally tractable. This algorithm is described by the following steps:

1. The algorithm solves the discrete Lyapunov equation (5.27) to compute the matrix $\boldsymbol{P} \in \mathcal{S}_n^+$. This solution can be approximated. To solve this equation, the recommended method depends on the dimension of the problem. With 4.4a small dimension, one can use matrix factorisation as in [34]. In high dimensions, one can use a bilinear transformation as described in [28].

2. The algorithm computes $\mathcal{P}_h\left(\mathscr{E}\right) = \mathcal{E}\left(\boldsymbol{\Gamma}_{out}\right)$ with the ellipsoid $\mathscr{E} = \mathcal{E}\left(10^{-\alpha}\boldsymbol{P}^{-\frac{1}{2}}\right)$ with a scale factor $\alpha > 0$.

3. The algorithm tests the inclusion (5.18) with the ellipsoid. If the inclusion is not verified, the ellipsoid is shrunk with higher values of α, until the inclusion is verified. From Theorem 4.4, one has

$$\left(\mathcal{P}_h\left(\mathcal{E}\left(\boldsymbol{\Gamma}\right)\right) \subset \mathcal{E}\left(\boldsymbol{\Gamma}\right)\right) \Leftrightarrow \left(\boldsymbol{\Gamma}_{out}^{-2} - \boldsymbol{\Gamma}^{-2} > 0\right) \tag{5.38}$$

   This inclusion can be numerically guaranteed by making a Cholesky decomposition of $\boldsymbol{\Gamma}_{out}^{-2} - \boldsymbol{\Gamma}^{-2}$. If the decomposition succeed, then $\boldsymbol{\Gamma}_{out}^{-2} - \boldsymbol{\Gamma}^{-2}$ is positive definite. However, if the decomposition fails, the algorithm is not able to conclude on the inclusion.

To compute the enclosure of the Jacobian on the ellipsoid $[\boldsymbol{J}]$ from the analytical expression of $\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}$, one can deduce an analytical expression of an inclusion function $\left[\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}\right]$ by substituting variables with intervals. One can then compute $[\boldsymbol{J}]$ as

$$[\boldsymbol{J}] = \left[\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}\right]\left([\mathscr{E}]\right), \tag{5.39}$$

with the box

$$[\mathscr{E}] = \mathrm{diag}(\left\|\boldsymbol{\Gamma}_1\right\|_2, \left\|\boldsymbol{\Gamma}_2\right\|_2, ..., \left\|\boldsymbol{\Gamma}_n\right\|_2) \cdot [1_n], \tag{5.40}$$

that enclose the ellipsoid, where $\boldsymbol{\Gamma}_i$ is the i^{th} column of $\boldsymbol{\Gamma}$.

In the algorithm, the most complex operations are matrix inversion and matrix square root which have polynomial complexity. In practice, Algorithm 2 is implemented in Python language with the libraries *Numpy* for matrix operations, *Codac* for Interval analysis and *Scipy* for solving Lyapunov equations.

Discussion The success of the algorithm depends on the opposite effect of $\boldsymbol{J}$ and ρ: while $\boldsymbol{J}$ contracts the ellipsoid, the pessimism in ρ increase the size of $\mathcal{P}_h\left(\mathscr{E}\right)$. To succeed, ρ must be small enough. Its value is affected by

- The dimension of the problem. To recall, in the algorithm 1 that computes $\mathcal{P}_h\left(\mathscr{E}\right)$, ρ is computed as the upper bound of the norm-2 of a box

$$\begin{aligned} \rho &= \sup\left\{\left\|\left[\widetilde{\boldsymbol{b}}\right]\right\|\right\}, \\ &= \sup\left\{\sqrt{\sum_{i=1}^{n}\left[\widetilde{b}_i\right]^2}\right\}. \end{aligned} \tag{5.41}$$

The norm-2 is computed by summing the square of the interval of each dimension of $\left[\widetilde{\boldsymbol{b}}\right]$. Thus the pessimism on each dimension piles up on the computation of ρ.

- The non-linearity of the system. Whereas a linear system gives no pessimism ($\rho = 0$) the more nonlinear the system, the bigger the gain.
- The size of the ellipsoid $\mathscr{E}$. The algorithm progressively reduces the size of $\mathscr{E}$ which reduces the nonlinear pessimism. Thus, $\mathcal{P}_h(\mathscr{E})$ converge towards $\boldsymbol{h}(\mathscr{E})$ and ρ converge towards 0.

Therefore, in theory, the algorithm always succeeds when $\mathscr{E}$ is small enough. However, in practice, numerical computations cannot be made on infinitely small ellipsoids and the value of ρ often converges towards a non-null value. So after some attempts with different values of α, one can consider that the method has failed. A good conditioning of the system could help to reduce pessimism and avoid failure.

Then, although the analytical expression of $\boldsymbol{h}$ is know in this chapter, the algorithm can also be used when $\boldsymbol{h}$ is the flow of a continuous-time dynamical system. Indeed, the algorithm only needs to approximate the Jacobian matrix $\boldsymbol{J}$ and enclose the Jacobian matrix on the ellipsoid with the interval matrix $[\boldsymbol{J}]$. This can be done using the guaranteed integration of the variational equation, presented in Chapter 3. As a result, the algorithm could be used to study continuous or hybrid systems, by discretising the system. the resulting evolution functions $\boldsymbol{h}$ may have no analytical expressions it is possible to evaluate their Jacobian matrices. This idea will be investigated in the following chapters.

5.6 Application

This section presents two applications of the Algorithm 2. It is first applied on a 2d pendulum. Then, the algorithm is used on a formation control with two robots. This second example is also used in the following chapters.

5.6.1 2d simple pendulum example

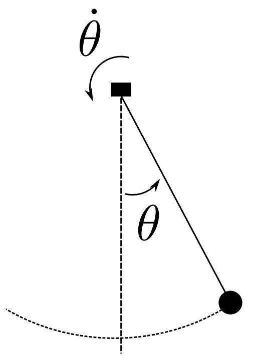


Figure 5.5: Simple pendulum

In this example, consider a simple pendulum illustrated by Figure 5.5, with the angle $\theta(t)$. Assume that the continuous-time dynamics of the pendulum is

$$\ddot{\theta}(t) = -\sin(\theta(t)) - \dot{\theta}(t), \tag{5.42}$$

where $\dot{\theta}$ is the angular speed and $\ddot{\theta}$ is the angular acceleration. Although the motion of the pendulum is described by a continuous-time dynamical system, this system can be discretised by an Euler scheme into the auxiliary discrete-time nonlinear system

$$\begin{aligned} \boldsymbol{m}_{k+1} &= \boldsymbol{h}(\boldsymbol{m}_k) \\ &= \boldsymbol{m}_k + d_t \cdot \begin{bmatrix} m_{2,k} \\ -\sin(m_{1,k}) - m_{2,k} \end{bmatrix}, \end{aligned} \tag{5.43}$$

with the discretisation time $d_t > 0$ and where $m_{1,k} = \theta(k \cdot d_t)$, $m_{2,k} = \dot{\theta}(k \cdot d_t)$. Assume that $d_t = 0.5\,\mathrm{s}$. In this example, the stability of 5.43 is analysed with Algorithm 2. As later presented in Chapter 6, the stability of the continuous-time system can be deduced from the stability of the auxiliary discrete-time system.

Note that, in this example, one can also prove the stability of this system by finding a Lyapunov function as in example 3.4. However, for the group of robots which are in high dimensions, Lyapunov functions are difficult to find.

5.6.1.1 Application of Algorithm 2

Choosing the ellipsoidal candidate. The algorithm starts by computing the Jacobian matrix at the origin

$$\begin{aligned} \boldsymbol{J} &= \frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}(\boldsymbol{0}) \\ &= \begin{bmatrix} 1 & d_t \\ -d_t & 1 - d_t \end{bmatrix}. \end{aligned} \tag{5.44}$$

Then, the algorithm solves the axis-aligned discrete Lyapunov equation

$$\boldsymbol{J}^T \boldsymbol{P} \boldsymbol{J} - \boldsymbol{P} = -\boldsymbol{J}^T \boldsymbol{J}. \tag{5.45}$$

From example 5.1, the solution is

$$\boldsymbol{P} = \begin{bmatrix} \frac{59}{13} & \frac{28}{13} \\ \frac{28}{13} & \frac{47}{13} \end{bmatrix}.$$

Then, with $\alpha = 0$, the first candidate ellipsoid is $\mathcal{E}(\boldsymbol{\Gamma})$ with

$$\begin{aligned} \boldsymbol{\Gamma} &= \boldsymbol{P}^{-\frac{1}{2}} \\ &\simeq \begin{bmatrix} 0.53 & -0.16 \\ -0.16 & 0.60 \end{bmatrix}. \end{aligned} \tag{5.46}$$

Propagation of the ellipsoid. Then, using Algorithm 1, the ellipsoid $\mathcal{E}(\boldsymbol{\Gamma})$ is propagated with the mapping $\boldsymbol{h}$ and the outer enclosure $\mathcal{E}(\boldsymbol{\Gamma}_{\text{out}}) = \mathcal{P}_h(\mathcal{E}(\boldsymbol{\Gamma}))$ is computed. For this propagation, the box

$$\begin{aligned}[\mathcal{E}] &= \begin{bmatrix} \|\boldsymbol{\Gamma}_1\|_2 & 0 \\ 0 & \|\boldsymbol{\Gamma}_2\|_2 \end{bmatrix} \cdot [\mathbf{1}]_n\,, \\ &\simeq \begin{bmatrix} [-0.56, 0.56] \\ [-0.62, 0.62] \end{bmatrix}, \end{aligned} \tag{5.47}$$

that enclose $\mathcal{E}(\boldsymbol{\Gamma})$ is computed with $\boldsymbol{\Gamma}_1$ and $\boldsymbol{\Gamma}_2$ the lines of $\boldsymbol{\Gamma}$. As illustrated on Figure 5.6, one has $\mathcal{E}(\boldsymbol{\Gamma}) \subseteq [\mathcal{E}]$. Then the Jacobian of $\boldsymbol{h}$ is evaluated on $\mathcal{E}$ with the interval matrix

$$\begin{aligned}[\boldsymbol{J}] &= \left[\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}\right]([\mathcal{E}]) \\ &= \begin{bmatrix} [1] & [d_t] \\ -d_t \cdot \cos([\mathcal{E}]_1) & [1 - d_t] \end{bmatrix} \\ &\simeq \begin{bmatrix} [1] & [0.5] \\ [-0.5, -0.42] & [0.5] \end{bmatrix}, \end{aligned} \tag{5.48}$$

where $[\mathcal{E}]_1$ is the first interval of $[\mathcal{E}]$ and such that $\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}(\mathcal{E}) \subseteq [\boldsymbol{J}]$. In the next step, the box

$$\begin{aligned}\left[\widetilde{\boldsymbol{b}}\right] &= \left(\boldsymbol{\Gamma}^{-1} \cdot \boldsymbol{J}^{-1} \cdot [\boldsymbol{J}] \cdot \boldsymbol{\Gamma} - \boldsymbol{I}_n\right) \cdot [\mathbf{1}]_n \\ &\simeq \begin{bmatrix} [-0.033, 0.033] \\ [-0.11, 0.11] \end{bmatrix} \end{aligned} \tag{5.49}$$

is computed, to obtain the inflation gain

$$\begin{aligned}\rho &= \sup\left\{\left\|\left[\widetilde{\boldsymbol{b}}\right]\right\|\right\} \\ &\simeq 0.11, \end{aligned} \tag{5.50}$$

to get the shape matrix of the outer enclosure

$$\begin{aligned}\boldsymbol{\Gamma}_{\text{out}} &= (1+\rho)\left(\boldsymbol{J}\boldsymbol{\Gamma}^2\boldsymbol{J}^T\right)^{\frac{1}{2}} \\ &\simeq \begin{bmatrix} 0.51 & -0.12 \\ -0.12 & 0.56 \end{bmatrix}. \end{aligned} \tag{5.51}$$

Inclusion of the ellipsoids. Then, the algorithm tests the inclusion $\mathcal{E}(\boldsymbol{\Gamma}_{\text{out}}) \subset \mathcal{E}(\boldsymbol{\Gamma})$ by computing

$$\begin{aligned}\boldsymbol{M} &= \boldsymbol{\Gamma}_{out}^{-2} - \boldsymbol{\Gamma}^{-2}, \\ &\simeq \begin{bmatrix} -0.05 & -0.4 \\ -0.4 & 0.12 \end{bmatrix}. \end{aligned}$$

However, $\boldsymbol{M}$ is not positive invariant, therefore the inclusion is not verified, as illustrated by Figure 5.6.

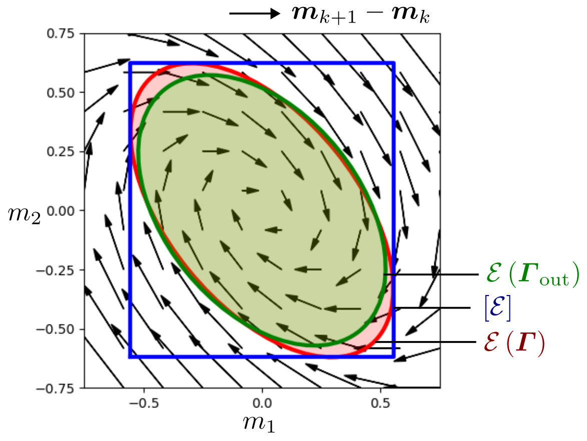


Figure 5.6: The propagated ellipsoid $\mathcal{E}(\boldsymbol{\Gamma}_{\text{out}})$ not included in the candidate $\mathcal{E}(\boldsymbol{\Gamma})$. The algorithm 2 continues with a smaller candidate.

Propagation and inclusion with a smaller candidate. To reduce the pessimism, the algorithm then test the ellipsoid $\mathcal{E}\left(\boldsymbol{\Gamma}'\right)$ with

$$\begin{aligned}\boldsymbol{\Gamma}' &= 10^{-\alpha} \cdot \boldsymbol{P}^{-\frac{1}{2}} \\ &\simeq \begin{bmatrix} 0.16 & -0.051 \\ -0.051 & 0.19 \end{bmatrix},\end{aligned}$$

with $\alpha = 1$. It computes the propagated ellipsoid $\mathcal{E}\left(\boldsymbol{\Gamma}'_{\text{out}}\right) = \mathcal{P}_h\left(\mathcal{E}\left(\boldsymbol{\Gamma}'\right)\right)$ with

$$\begin{aligned}\boldsymbol{\Gamma}'_{\text{out}} &= \left(1+\rho'\right)\left(\boldsymbol{J}\boldsymbol{\Gamma}'^2\boldsymbol{J}^T\right)^{\frac{1}{2}} \\ &\simeq \begin{bmatrix} 0.15 & -0.034 \\ -0.034 & 0.16 \end{bmatrix},\end{aligned} \tag{5.52}$$

with a smaller inflation gain

$$\rho' \simeq 0.011. \tag{5.53}$$

Then, the ellipsoid

$$\begin{aligned}\boldsymbol{M}' &= \boldsymbol{\Gamma}'^{-2}_{out} - \boldsymbol{\Gamma}'^{-2} \\ &\simeq \begin{bmatrix} 8.7 & -0.48 \\ -0.48 & 8.9 \end{bmatrix},\end{aligned} \tag{5.54}$$

is verified positive invariant with a Cholesky decomposition. Thus $\mathcal{E}\left(\boldsymbol{\Gamma}'_{\text{out}}\right) \subset \mathcal{E}\left(\boldsymbol{\Gamma}'\right)$ is verified, as illustrated by Figure 5.7. Therefore, the result of the algorithm is the ellipsoid $\mathcal{E}\left(\boldsymbol{\Gamma}'\right)$ that verifies $\mathcal{P}_h\left(\mathcal{E}\left(\boldsymbol{\Gamma}'\right)\right) \subset \mathcal{E}\left(\boldsymbol{\Gamma}'\right)$. As a result, from Theorem 5.1, $\mathcal{E}\left(\boldsymbol{\Gamma}'\right)$ is positive invariant with respect to the system and the system is locally exponentially stable on the ellipsoid $\mathcal{E}\left(\boldsymbol{\Gamma}'\right)$.

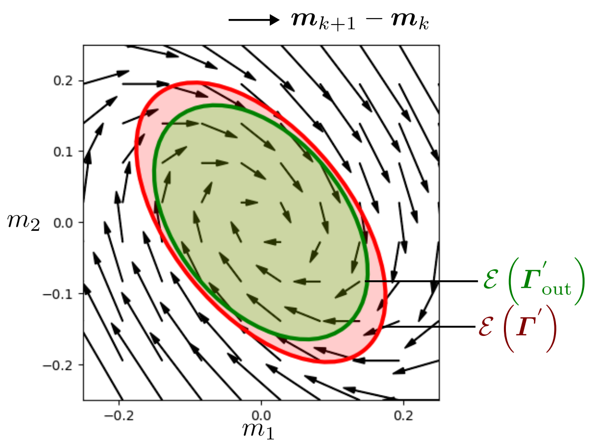


Figure 5.7: The propagated ellipsoid $\mathcal{E}\left(\boldsymbol{\Gamma}'_{\text{out}}\right)$ is included in the candidate $\mathcal{E}\left(\boldsymbol{\Gamma}'\right)$. The algorithm 2 succeeds.

5.6.2 Formation control with two ROVs

In this example, two ROVs are controlled with a virtual structure approach to reach an equilateral triangular formation, as illustrated in Figure 5.8. One vertex of the triangle is fixed and serves as the reference point. The two other vertices are the desired position of the ROVs. As a result, the triangle can pivot around the reference point. This degree of freedom is not controlled. This formation can be useful when the ROVs must inspect a structure or patrol around the pivot point. Moreover, if a human operator takes control of one robot, the other one will follow it.

In this system, the source of non-linearity comes from a change in the coordinate reference. The formation is defined in polar coordinates. However, the position-tracking controller of the robot is designed in Cartesian coordinates, as is often the case in practice.

The blue robot is called *Inky*, and the red robot is called *Blinky*. These are the names of the real robots used in Chapter 9. Their name is a reference to the *Pac-Man* video game.

In this chapter, the system is modelled by a discrete-time linear system. This system is obtained by an Euler discretisation of the real continuous-time dynamics of the robots. This discretisation will be useful in the application of Chapter 6 to compute a positive invariant ellipsoid. Moreover, the following chapters will consider other types of models with continuous-time behaviours.

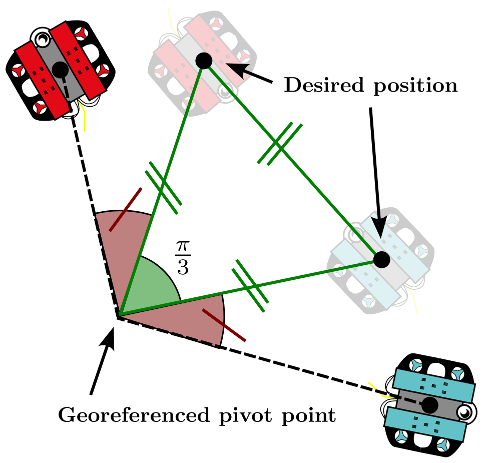


Figure 5.8: Equilateral triangular formation with *Inky* (blue) and *Blinky* (red). The pivot point is fixed on Earth.

5.6.2.1 Mathematical description of the system

As illustrated by Figure 5.9, the polar horizontal coordinates of the ROVs are written

$$(d_{b,k}, \phi_{b,k}) \in \mathbb{R}^2 (\text{for Inky}),$$
$$(d_{r,k}, \phi_{r,k}) \in \mathbb{R}^2 (\text{for Blinky}),$$

and their polar speed is written

$$(v_{b,k}, w_{b,k}) \in \mathbb{R}^2 (\text{for Inky}),$$
$$(v_{r,k}, w_{r,k}) \in \mathbb{R}^2 (\text{for Blinky}).$$

Note that $d_{b,k}$ is a distance, $v_{b,k}$ is a speed, $\phi_{b,k}$ is an angle and $w_{b,k}$ is an angular velocity around the origin. The motion robots are described by the recursive formula

$$\begin{cases} d_{b,k+1} &= d_{b,k} + T \cdot v_{b,k}, \\ d_{r,k+1} &= d_{r,k} + T \cdot v_{r,k}, \\ \phi_{b,k+1} &= \phi_{b,k} + T \cdot w_{b,k}, \\ \phi_{r,k+1} &= \phi_{r,k} + T \cdot w_{r,k}, \\ v_{b,k+1} &= v_{b,k} + T \cdot u_{1,k}, \\ v_{r,k+1} &= v_{r,k} + T \cdot u_{2,k}, \\ w_{b,k+1} &= w_{b,k} + T \cdot \frac{u_{3,k}}{d_{b,k}}, \\ w_{r,k+1} &= w_{r,k} + T \cdot \frac{u_{4,k}}{d_{r,k}}. \end{cases} \tag{5.55}$$

with a time step $T > 0$ and where $(u_{1,k}, u_{2,k}, u_{3,k}, u_{4,k}) \in \mathbb{R}^4$ are the controlled acceleration inputs of the system. This equation of motion has a singularity when $d_{b,k} = 0$ or when $d_{r,k} = 0$. The singular points are avoided during the stability analysis, as they are not part of the ellipsoids. Then, as illustrated by Figure 5.9, the desired

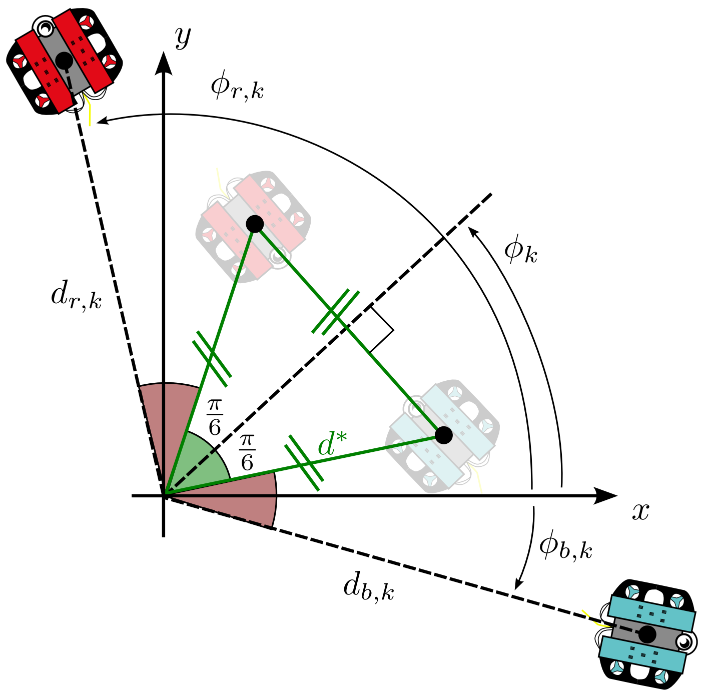


Figure 5.9: Definition of the desired formation

positions of the robot are

$$\left(d^*, \phi_k - \frac{\pi}{6}\right)^T \text{(for Inky)}, \tag{5.56}$$

$$\left(d^*, \phi_k + \frac{\pi}{6}\right)^T \text{(for Blinky)}, \tag{5.57}$$

with the desired distance $d^* > 0$ and where the orientation of the triangle ϕ_k is given by

$$\phi_k = \frac{\phi_{b,k} + \phi_{r,k}}{2}. \tag{5.58}$$

The robots track their desired position with the following saturated proportional derivative controller

$$\begin{aligned}
u_{1,k} &= s \cdot \arctan\left(k_{p,d} \cdot (d^* - d_{b,k}) - k_{d,d} \cdot v_{b,k}\right), \\
u_{2,k} &= s \cdot \arctan\left(k_{p,d} \cdot (d^* - d_{r,k}) - k_{d,d} \cdot v_{r,k}\right), \\
u_{3,k} &= s \cdot \arctan\left(k_{p,\phi} \cdot \left(\phi_k - \frac{\pi}{6} - \phi_{b,k}\right) - k_{d,\phi} \cdot w_{b,k}\right), \\
u_{4,k} &= s \cdot \arctan\left(k_{p,\phi} \cdot \left(\phi_k + \frac{\pi}{6} - \phi_{r,k}\right) - k_{d,\phi} \cdot w_{r,k}\right),
\end{aligned} \tag{5.59}$$

with the saturation amplitude $s > 0$ and the controller gains $(k_{p,d}, k_{p,\phi}, k_{d,d}, k_{d,\phi}) > 0$. Finally, to describe the full system, consider the global state vector $\boldsymbol{m}_k =$

$(m_{i,k})_{i\in[\![1:6]\!]} \in \mathbb{R}^6$ with

$$\begin{aligned}
m_{1,k} &= d_{b,k} - d^* \\
m_{2,k} &= d_{r,k} - d^* \\
m_{3,k} &= \phi_{r,k} - \phi_{b,k} - \frac{\pi}{3} \\
m_{4,k} &= v_{b,k} \\
m_{5,k} &= v_{r,k} \\
m_{6,k} &= w_{b,k} \\
m_{7,k} &= w_{r,k}
\end{aligned} \tag{5.60}$$

The evolution of the global state vector is given by the mapping

$$\begin{aligned}
\boldsymbol{m}_{k+1} &= \boldsymbol{h}(\boldsymbol{m}_k), \\
\boldsymbol{0} &= \boldsymbol{h}(\boldsymbol{0}),
\end{aligned} \tag{5.61}$$

with

$$\boldsymbol{h}(\boldsymbol{m}_k) = \begin{bmatrix}
m_{1,k} + T \cdot m_{4,k} \\
m_{2,k} + T \cdot m_{5,k} \\
m_{3,k} + T \cdot (m_{7,k} - m_{6,k}) \\
m_{4,k} + sT \cdot \arctan(-k_{p,d} \cdot m_{1,k} - k_{d,d} \cdot m_{4,k}) \\
m_{5,k} + sT \cdot \arctan(-k_{p,d} \cdot m_{2,k} - k_{d,d} \cdot m_{5,k}) \\
m_{6,k} + \frac{sT}{m_{1,k}+d^*} \arctan\left(\frac{k_{p,\phi}}{2} m_{3,k} - k_{d,\phi} \cdot m_{6,k}\right) \\
m_{7,k} + \frac{sT}{m_{2,k}+d^*} \arctan\left(-\frac{k_{p,\phi}}{2} m_{3,k} - k_{d,\phi} \cdot m_{7,k}\right)
\end{bmatrix}. \tag{5.62}$$

The computation of $\boldsymbol{h}(\boldsymbol{m}_k)$ is detailed in Appendix 10. Moreover, by differentiating $\boldsymbol{h}$, the Jacobian matrix of this evolution function is

$$\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}(\boldsymbol{m}) = \begin{bmatrix}
1 & 0 & 0 & T & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & T & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & -T & T \\
J_{4,1} & 0 & 0 & J_{4,4} & 0 & 0 & 0 \\
0 & J_{5,2} & 0 & 0 & J_{5,5} & 0 & 0 \\
J_{6,1} & 0 & J_{6,3} & 0 & 0 & J_{6,6} & 0 \\
0 & J_{7,2} & J_{7,3} & 0 & 0 & 0 & J_{7,7}
\end{bmatrix} \tag{5.63}$$

where the expression of the $J_{i,j}$ are detailed in Appendix 10. Thus, the matrix $\boldsymbol{J} = \frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}(\boldsymbol{0})$ can be computed with (5.63). The inclusion function $\left[\frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}\boldsymbol{m}}\right]$ is also deduced from the analytical expression (5.63).

5.6.2.2 Stability analysis

The values of the parameters are detailed in Table 5.1. With these values, the eigenvalues of $\boldsymbol{J}$ are all in the unit circle so that $\boldsymbol{J}$ is a Schur matrix. So, from Theorem 3.8, the system is asymptotically stable at the origin. However, because of the controller saturation and the singularity in the evolution function, the system is likely not globally asymptotically stable.

Then the Algorithm 2 results in a 7-dimensional ellipsoid $\mathscr{E}$ that verifies

$$\mathcal{P}_h\left(\mathscr{E}\right) \subset \mathscr{E}. \tag{5.64}$$

Thus, from Theorem 5.1, $\mathscr{E}$ is positive invariant with respect to the system and the system is locally exponentially stable on the ellipsoid $\mathscr{E}$.

A high-dimensional inclusion is not easily represented on a 2-dimensional paper. Nevertheless, using orthogonal projections, Figure 5.10 illustrates the ellipsoids $\mathscr{E}$ and $\mathcal{P}_h\left(\mathscr{E}\right)$ and their inclusion. On this figure, one can see that the ellipsoids have the same semi-axis, as explained in Section 5.4.

Moreover, the ellipsoid $\mathcal{P}_h\left(\mathscr{E}\right)$ is just slightly smaller than $\mathscr{E}$, even if the value of the inflation gain was about $\rho \simeq 0.009$. This can be explained, by the fact that the norm of the eigenvalues of $\boldsymbol{J}$ are in the interval $[0.85, 0.96]$ an thus close to 1. Thus, the inclusion (5.64) can only be verified with little pessimistic inflation. Moreover, the inclusion (5.64) was not verified at the first iteration of α which shows that one has to study smaller ellipsoids.

Parameter	value	unit
T	0.1	s
s	10	m.s^{-2}
$k_{p,d}$	0.1	m^{-1}
$k_{d,d}$	0.1	s.m^{-1}
$k_{p,\phi}$	0.2	rad^{-1}
$k_{d,\phi}$	0.7	s.rad^{-1}
d^*	5	m

Table 5.1: Parameters of the system

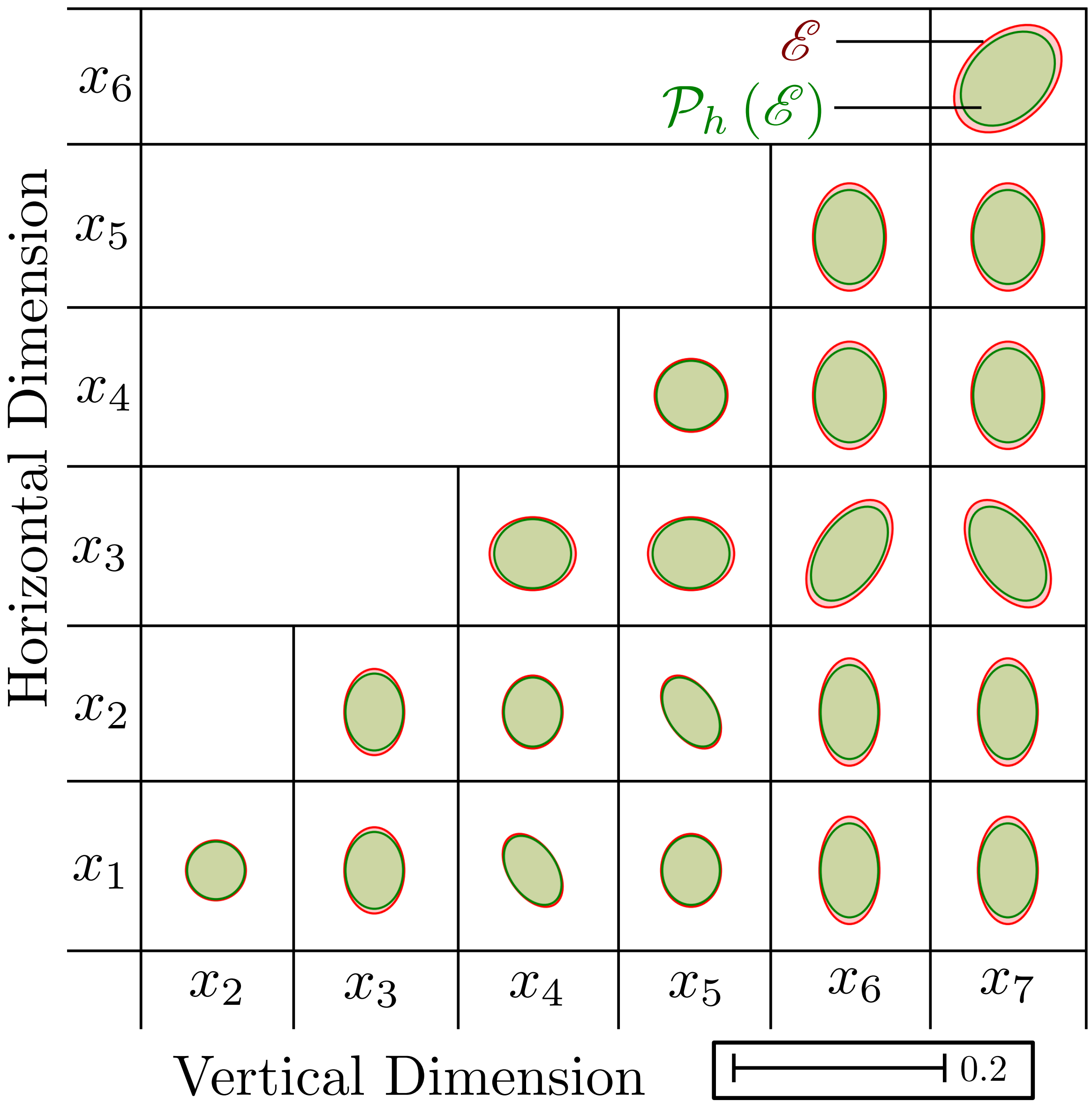


Figure 5.10: Representation of the 7-dimensional ellipsoids $\mathscr{E}$ (red) and $\mathcal{P}_h(\mathscr{E})$ (green) with orthogonal projections on the 2-dimensional planes $(0, x_i, x_j)$ with $i < j$.

5.6.2.3 Simulation of the system

To illustrate the positive invariance of the computed ellipsoid $\mathscr{E}$, the system is simulated with a random initial state $\boldsymbol{m}_0 \in \mathscr{E}$. The result of this simulation is shown illustrated by Figure 5.11. At each step of the simulation, $\boldsymbol{m}_k \in \mathscr{E}$ is verified. As demonstrated by the previous section, one can observe that every state variable converges to zero.

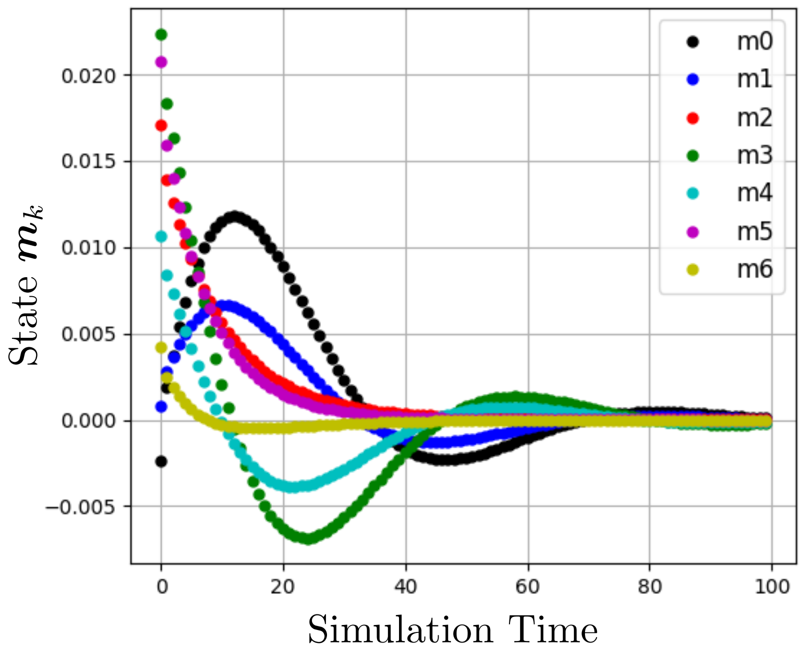


Figure 5.11: Simulation of the state $\boldsymbol{m}_k$ with a period T and a simulation time $t_k = k \cdot T$, with the parameters of Table 5.1.

5.7 Conclusion

This chapter presents a numerical method to compute a positive invariant ellipsoid with respect to a high-dimensional nonlinear discrete-time dynamical system such that the system is exponentially stable in the ellipsoid. To our knowledge, this is the first guaranteed numerical method that can compute an ellipsoidal domain of attraction for a high-dimensional nonlinear system. This method can also be used when the analytical expression of the evolution function of the discrete system is unknown, as long as one can provide a function that encloses the values of the Jacobian matrix. Moreover, the computations are guaranteed by interval analysis.

Therefore, This method can be applied to high-dimensional problems, where Lyapunov functions are difficult to find. In the following chapters, this method will be adapted to study continuous and hybrid nonlinear systems.

Moreover, the method assumes that the mapping of the systems is non-singular, which is the case for discrete-time systems most of the time. The singular case is discussed in Chapter 7.

The results of this chapter have been submitted to IEEE Transactions on Automatic Control [63].

Chapter 6

A guaranteed numerical method to find ellipsoidal domain of attraction for nonlinear continuous-time system

6.1 Introduction

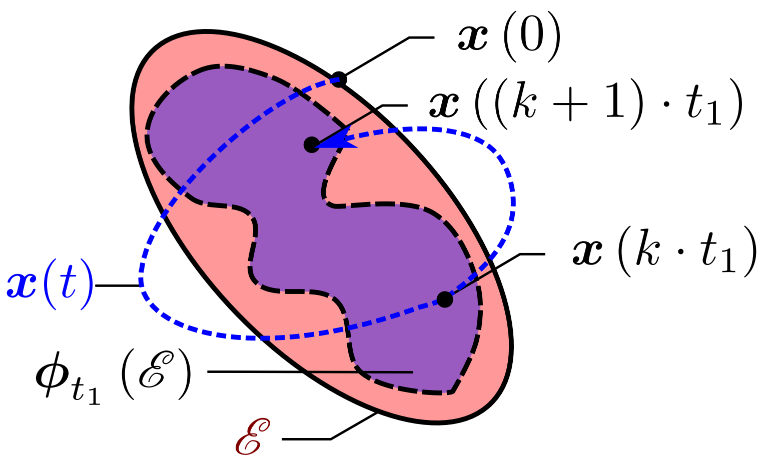
Chapter 5 presented a numerical method to find a positive invariant ellipsoid in which a discrete nonlinear system is exponentially stable. As robots are also composed of continuous dynamics, the method must be adapted to continuous-time systems. This adaptation can be presented in two points

The first method of Chapter 5 can be applied to a discretisation of the continuous system. As explained in Section 5.5, the Jacobian matrix of the discretised mapping can be evaluated with a guaranteed integration of the variational equation. This chapter shows that if the discretised system is proved exponentially stable in an ellipsoid $\mathscr{E}$ with this method, then the continuous system is also exponentially stable in $\mathscr{E}$.

Second, while this approach proves that $\mathscr{E}$ is positive invariant with respect to the discretised system, it only proves that $\mathscr{E}$ is p-invariant (or periodic invariant) with respect to the continuous system, as illustrated by Figure 6.1, where the state comes back periodically inside the ellipsoid of origin. Therefore, this chapter also presents an alternative method to prove that $\mathscr{E}$ is positive invariant with respect to the continuous system. This method consist in propagating $\mathscr{E}$ with an Euler scheme.

As in Chapter 3.2.2, these new methods are based on the guaranteed propagation of ellipsoids. Thus they are computationally tractable and can be used on high dimensional systems. To our knowledge, these are the first guaranteed numerical methods that can compute an ellipsoidal domain of attraction for a high-dimensional continuous-time nonlinear system.

This chapter is organised as follows. Section 6.2 presents the stability problem considered in this chapter. Section 6.3 shows a method to find a positive invariant ellipsoid for a continuous system using ellipsoidal propagation with an Euler Scheme. Section 6.4 presents how the exponential stability of the continuous system is deduced



$\boldsymbol{x}(0)$
$\boldsymbol{x}((k+1)\cdot t_1)$
$\boldsymbol{x}(k \cdot t_1)$
$\boldsymbol{x}(t)$
$\boldsymbol{\phi}_{t_1}(\mathscr{E})$
$\mathscr{E}$

Figure 6.1: Example of a p-invariant ellipsoid $\mathscr{E}$, where the state $\boldsymbol{x}$ follows a flow $\boldsymbol{\phi}$. The ellipsoid $\mathscr{E}$ is positive invariant with respect to the discretised system $\boldsymbol{x}((k+1)\cdot t_1) = \boldsymbol{\phi}_{t_1}(\boldsymbol{x}(k \cdot t_1))$. However, the state may escape the ellipsoid between the discrete times $k \cdot t_1$.

from the stability of the discretised system. Section 6.5 present an adaptation of Algorithm 2 from Chapter 5 to continuous systems. Finally, Section 6.6 shows an application of the method on a continuous version of the formation control problem of Section 5.2.

6.2 Problem definition

In this Chapter, consider a continuous nonlinear dynamical system described by the following ODE

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t)), \tag{6.1}$$

where $t \in \mathbb{R}$ represents the time, $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a $K-$Lipschitz nonlinear function and $\boldsymbol{x}(t) \in \mathbb{R}^n$ represents the state of the robot at the time t. Assume that the analytical expression of $\boldsymbol{f}$ is known. The system is represented such that the origin $\mathbf{0}$ of $\mathbb{R}^n$ is an equilibrium point, $\boldsymbol{f}(\mathbf{0}) = \mathbf{0}$. The system is assumed locally exponentially stable, so the Jacobian at the origin

$$\boldsymbol{A} = \frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{x}}(\mathbf{0}) \tag{6.2}$$

has eigenvalues with only strictly negative real parts.

For continuous systems, the *Nagumo theorem* (Theorem 6.1) gives a general criterion for the positive invariance of ellipsoids.

Theorem 6.1. *[6, Section 4.4.1] The ellipsoid $\mathscr{E}$ is PI w.r.t. (6.1) if and only if*

$$\forall \boldsymbol{x}_0 \in \partial\mathscr{E}, \langle \boldsymbol{x}_0, \boldsymbol{f}(\boldsymbol{x}_0)\rangle_{\boldsymbol{\Gamma}^{-2}} \leq 0, \tag{6.3}$$

with $\boldsymbol{\Gamma} = \mathcal{E}^{-1}(\mathscr{E})$.

Remark 6.1. Sketch of a proof. Behind (6.3), there is the quadratic positive function

$$\begin{aligned} V :& \mathbb{R}^n \to \mathbb{R}^n, \\ & \boldsymbol{x}_0 \mapsto \|\boldsymbol{x}_0\|_{\boldsymbol{\Gamma}^{-2}}, \end{aligned} \tag{6.4}$$

whose time derivative is

$$\dot{V}\left(\boldsymbol{x}\left(t\right)\right) = 2\left\langle \boldsymbol{x}\left(t\right), \boldsymbol{f}\left(\boldsymbol{x}\left(t\right)\right)\right\rangle_{\boldsymbol{\Gamma}^{-2}}. \tag{6.5}$$

and which acts like a Lyapunov function on the border of the ellipsoids $\partial\mathscr{E}$. From the ellipsoid Definition 4.6, one has

$$\mathscr{E} = \left\{\boldsymbol{x}_0 \in \mathbb{R}^n | V\left(\boldsymbol{x}_0\right) \leq 1\right\}, \tag{6.6}$$

$$\partial\mathscr{E} = \left\{\boldsymbol{x}_0 \in \mathbb{R}^n | V\left(\boldsymbol{x}_0\right) = 1\right\}. \tag{6.7}$$

So a trajectory $\boldsymbol{x}\left(t\right)$ stays in $\mathscr{E}$ if and only if $\dot{V}\left(\boldsymbol{x}\left(t\right)\right) < 0$ when $\boldsymbol{x}\left(t\right) \in \partial\mathscr{E}$. However, $\dot{V}$ must not necessarily be negative inside the ellipsoid.

In other words, the ellipsoid is positive invariant if and only if on the border of the ellipsoid, the vector $\boldsymbol{f}\left(\boldsymbol{x}\right)$ points inside of the ellipsoid or is tangent to the ellipsoid, as illustrated by Figure 6.2.

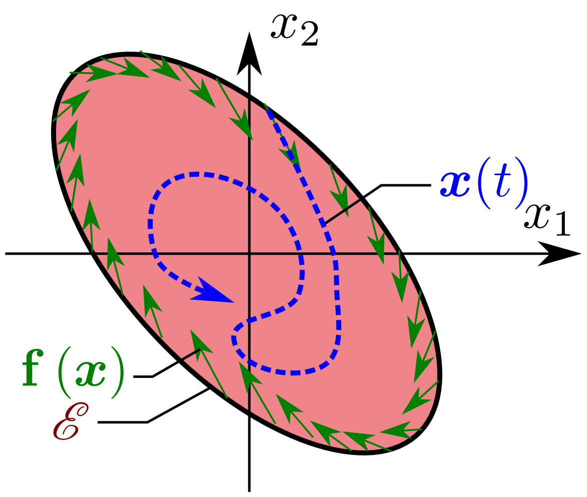


Figure 6.2: Illustration of the Nagumo Theorem for 2-dimensional ellipsoids. The vector field $\boldsymbol{f}\left(\boldsymbol{x}\right)$ prohibit the state trajectory $\boldsymbol{x}\left(t\right)$ from escaping the ellipsoid.

As in Section 5.2, the problem in this chapter consists in computing a positive invariant ellipsoid $\mathscr{E}$ included in the domain of attraction such that the system is locally exponentially stable in $\mathscr{E}$, i.e. there exist $\gamma \in]0,1[$ that verify

$$\boldsymbol{x}\left(t_0\right) \in \mathscr{E} \Rightarrow \|\boldsymbol{x}\left(t\right)\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha \|\boldsymbol{x}\left(t_0\right)\|_{\boldsymbol{\Gamma}^{-2}} e^{t\cdot\ln\gamma}, \forall t \in \mathbb{R}^+. \tag{6.8}$$

with $\boldsymbol{\Gamma} = \mathcal{E}^{-1}\left(\mathscr{E}\right)$. In this chapter, the problem of the positive invariance and the exponential stability are separated. Thus the problem is decomposed into two steps:

- Finding a positive invariant ellipsoid $\mathscr{E}$, as presented in Section 6.3.
- Proving that the system is exponentially stable in the positive invariant ellipsoid $\mathscr{E}$, as presented in Section 6.4.

6.3 Finding a positive invariant ellipsoid propagation

To test the inequality (6.3) of the Nagumo Theorem 6.2, there already exist some numerical methods such as the *SIVIA* algorithm [39]. Unfortunately, these methods have exponential complexity. Thus, to have a polynomial complexity, the inequality can be verified with Theorem 6.2.

Theorem 6.2. *Let $\mathscr{E}$ be a non-degenerate ellipsoid of $\mathbb{R}^n$. If there is a $\delta > 0$ so that $\mathscr{E}$ is PI w.r.t. the discrete system*

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \delta \cdot \boldsymbol{f}(\boldsymbol{x}_k), \tag{6.9}$$

then $\mathscr{E}$ is also PI with respect to the continuous dynamical system (6.1).

Proof. Let $\boldsymbol{\Gamma} = \mathcal{E}^{-1}(\mathscr{E})$, assume that $\mathscr{E}$ is PI w.r.t. the discrete system(6.9) with $\delta > 0$ and let $\boldsymbol{x}_0 \in \partial\mathscr{E}$. So, at the first iteration of the discrete system, the point

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \delta \cdot \boldsymbol{f}(\boldsymbol{x}_0) \tag{6.10}$$

is in $\mathscr{E}$. Therefore, from the definition of non-degenerate ellipsoids, one deduces that

$$\begin{aligned} &\|\boldsymbol{x}_1\|_{\boldsymbol{\Gamma}^{-2}} && \leq 1 \\ \Leftrightarrow &\langle \boldsymbol{x}_1, \boldsymbol{x}_1 \rangle_{\boldsymbol{\Gamma}^{-2}} && \leq 1. \end{aligned} \tag{6.11}$$

Moreover, one has

$$\begin{aligned} \langle \boldsymbol{x}_1, \boldsymbol{x}_1 \rangle_{\boldsymbol{\Gamma}^{-2}} &= \langle \boldsymbol{x}_0 + \delta \cdot \boldsymbol{f}(\boldsymbol{x}_0), \boldsymbol{x}_0 + \delta \cdot \boldsymbol{f}(\boldsymbol{x}_0) \rangle_{\boldsymbol{\Gamma}^{-2}} \\ &= \langle \boldsymbol{x}_0, \boldsymbol{x}_0 \rangle_{\boldsymbol{\Gamma}^{-2}} + 2\delta \langle \boldsymbol{x}_0, \boldsymbol{f}(\boldsymbol{x}_0) \rangle_{\boldsymbol{\Gamma}^{-2}} + \delta^2 \langle \boldsymbol{f}(\boldsymbol{x}_0), \boldsymbol{f}(\boldsymbol{x}_0) \rangle_{\boldsymbol{\Gamma}^{-2}} \\ &= \|\boldsymbol{x}_0\|^2_{\boldsymbol{\Gamma}^{-2}} + 2\delta \langle \boldsymbol{x}_0, \boldsymbol{f}(\boldsymbol{x}_0) \rangle_{\boldsymbol{\Gamma}^{-2}} + \delta^2 \|\boldsymbol{f}(\boldsymbol{x}_0)\|^2_{\boldsymbol{\Gamma}^{-2}}. \end{aligned} \tag{6.12}$$

Furthermore, since $\boldsymbol{x}_0 \in \partial\mathscr{E}$, one gets $\|\boldsymbol{x}_0\|^2_{\boldsymbol{\Gamma}^{-2}} = 1$. As a result, one obtains

$$\begin{aligned} 2\delta \langle \boldsymbol{x}_0, \boldsymbol{f}(\boldsymbol{x}_0) \rangle_{\boldsymbol{\Gamma}^{-2}} + \delta^2 \|\boldsymbol{f}(\boldsymbol{x}_0)\|^2_{\boldsymbol{\Gamma}^{-2}} &\leq 0 \\ \Leftrightarrow \langle \boldsymbol{x}_0, \boldsymbol{f}(\boldsymbol{x}_0) \rangle_{\boldsymbol{\Gamma}^{-2}} &\leq -\frac{\delta}{2} \|\boldsymbol{f}(\boldsymbol{x}_0)\|^2_{\boldsymbol{\Gamma}^{-2}} \\ \Rightarrow \langle \boldsymbol{x}_0, \boldsymbol{f}(\boldsymbol{x}_0) \rangle_{\boldsymbol{\Gamma}^{-2}} &\leq 0. \end{aligned}$$

Therefore, by Theorem 6.1, $\mathscr{E}$ is PI w.r.t. (6.1). □

This theorem considers a discretisation of the continuous dynamical system (6.1) by an Euler method with a time step $\delta > 0$. This discrete system (6.9) can be called an Euler auxiliary system. As illustrated by Figure 6.3, if $\mathscr{E}$ is positive invariant with respect the discrete system (6.9), then for all $\boldsymbol{x}_0 \in \partial\mathscr{E}$, one has $\boldsymbol{x}_1 \in \mathscr{E}$. As a result, the vector $\boldsymbol{f}(\boldsymbol{x}_0)$ points inside $\mathscr{E}$. So the Nagumo Theorem is verified. Therefore $\mathscr{E}$ is also PI w.r.t. the continuous dynamical system (6.1).

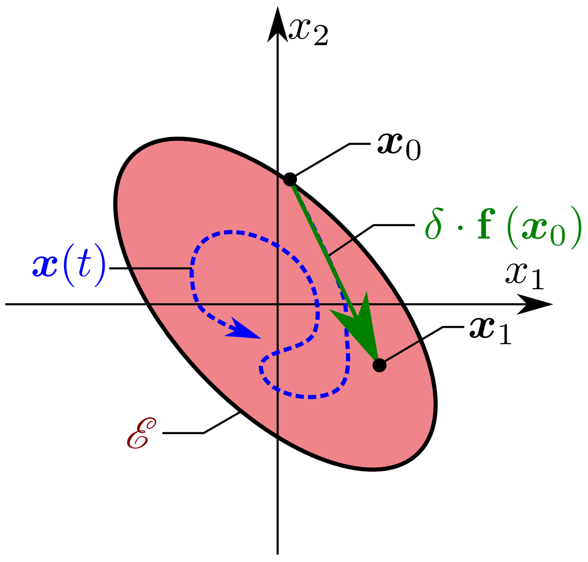


Figure 6.3: Intuition with an Euler method

Note that the reciprocity may not be verified if δ is too big. Even if $\boldsymbol{f}(\boldsymbol{x}_0)$ points inside the ellipsoid, $\delta \cdot \boldsymbol{f}(\boldsymbol{x}_0)$ may pierce through the ellipsoid and $\boldsymbol{x}_1$ may be outside the ellipsoid. Additional results on Euler auxiliary systems are presented in [6, Appendix A.1.].

The method of Chapter 5 can be used to verify that $\mathscr{E}$ is positive invariant with respect to the discrete system (6.9). The axis-aligned Lyapunov equation of Section 5.4 can be used to chose the ellipsoid $\mathscr{E}$ with the Jacobian matrix

$$\boldsymbol{J} = \boldsymbol{I}_n + \delta \cdot \frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}t}(\mathbf{0}) . \tag{6.13}$$

6.4 Deducing the exponential stability

In Section 6.3, the continuous system is discretised via an Euler scheme. To prove the exponential stability of the system, this section now considers the exact discretisation of the system

$$\boldsymbol{x}(t_{k+1}) = \boldsymbol{\phi}_T(\boldsymbol{x}(t_k)) \tag{6.14}$$

with the flow mapping $\boldsymbol{\phi}_T$ for a time $T > 0$. As presented in Theorem 6.3, if the discretised system is exponentially stable in an ellipsoid $\mathscr{E}$, then the continuous system is also exponentially stable in $\mathscr{E}$.

Theorem 6.3. *Consider a continuous time nonlinear dynamical system described by the following equations*

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t)), \\ \mathbf{0} &= \boldsymbol{f}(\mathbf{0}), \end{aligned} \tag{6.15}$$

*where $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a K-*Lipschitz *differentiable nonlinear function with $K > 0$. Let $T > 0$ and for all $k \in \mathbb{N}$ consider the time $t_k = k \cdot T$ and the discrete-time state vector*

$$\boldsymbol{x}_k = \boldsymbol{x}\left(t_k\right), \tag{6.16}$$

whose evolution is described by

$$\boldsymbol{x}_{k+1} = \boldsymbol{\phi}_t\left(\boldsymbol{x}_k\right), \tag{6.17}$$

where $\boldsymbol{\phi}_t$ is the flow function of the dynamical system. Let $\mathscr{E}$ be an ellipsoid of $\mathbb{R}^n$. If the discrete system (6.17) is exponentially stable in $\mathscr{E}$, then the dynamical system (6.15) is also locally exponentially stable in $\mathscr{E}$.

Proof. Consider the solution $\boldsymbol{x}$ of the continuous system (6.15) and let $\boldsymbol{\Gamma} = \mathcal{E}^{-1}\left(\mathscr{E}\right)$. From (5.4), since the discrete system is exponentially stable in $\mathscr{E}$, then there exist $\gamma \in]0,1[$ such that

$$\boldsymbol{x}\left(0\right) \in \mathscr{E} \Rightarrow \left\|\boldsymbol{x}\left(t_k\right)\right\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha \left\|\boldsymbol{x}\left(0\right)\right\|_{\boldsymbol{\Gamma}^{-2}} e^{k\cdot\ln\gamma}, \forall k \in \mathbb{N}, \tag{6.18}$$

Moreover, from Theorem 3.3, since $\boldsymbol{f}$ is K-Lipschitz, then for all $t \in [t_k, t_{k+1}]$ one gets

$$\left\|\boldsymbol{x}\left(t\right)\right\|_{\boldsymbol{\Gamma}^{-2}} \leq \left\|\boldsymbol{x}_k\right\|_{\boldsymbol{\Gamma}^{-2}} e^{K(t-t_k)}. \tag{6.19}$$

Let $k \in \mathbb{N}$ and $t \in [t_k, t_{k+1}]$. From (6.18) and (6.19), one obtains

$$\left\|\boldsymbol{x}\left(t\right)\right\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha \left\|\boldsymbol{x}_0\right\|_{\boldsymbol{\Gamma}^{-2}} e^{K(t-t_k)+k\ln\gamma}. \tag{6.20}$$

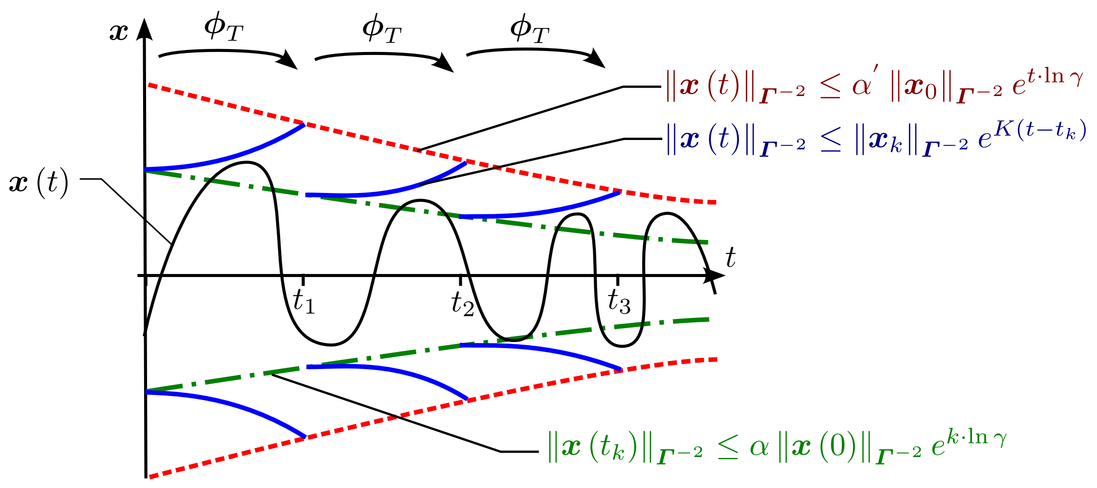
Then, with $\Delta t = t - t_k \in [0,T]$, one has

$$\begin{aligned} K\left(t-t_k\right) + k\ln\gamma &= K\left(t-t_k\right) + t_k \cdot \frac{\ln\gamma}{T}, \\ &= K\cdot\Delta t + \left(t-\Delta t\right)\frac{\ln\gamma}{T}, \\ &= \left(K - \frac{\ln\gamma}{T}\right)\Delta t + t\cdot\frac{\ln\gamma}{T}, \\ &< \left(K - \frac{\ln\gamma}{T}\right)T + t\cdot\frac{\ln\gamma}{T}. \end{aligned} \tag{6.21}$$

Thus, one deduces

$$\left\|\boldsymbol{x}\left(t\right)\right\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha^{'} \left\|\boldsymbol{x}_0\right\|_{\boldsymbol{\Gamma}^{-2}} e^{t\cdot\ln\gamma}, \tag{6.22}$$

with $\alpha^{'} = \alpha e^{\left(K+\frac{\ln\gamma}{T}\right)T} > 0$ and $\gamma^{'} = \gamma^{\frac{1}{T}} \in]0,1[$. Therefore, according to (6.8), the continuous system is exponentially stable in $\mathscr{E}$. $\square$



$\boldsymbol{x}$
$\boldsymbol{\phi}_T$
$\boldsymbol{\phi}_T$
$\boldsymbol{\phi}_T$
$\|\boldsymbol{x}(t)\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha' \|\boldsymbol{x}_0\|_{\boldsymbol{\Gamma}^{-2}} e^{t\cdot\ln\gamma}$
$\|\boldsymbol{x}(t)\|_{\boldsymbol{\Gamma}^{-2}} \leq \|\boldsymbol{x}_k\|_{\boldsymbol{\Gamma}^{-2}} e^{K(t-t_k)}$
$\boldsymbol{x}(t)$
t
t_1
t_2
t_3
$\|\boldsymbol{x}(t_k)\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha \|\boldsymbol{x}(0)\|_{\boldsymbol{\Gamma}^{-2}} e^{k\cdot\ln\gamma}$

Figure 6.4: Exponential convergence of the continuous state $\boldsymbol{x}(t)$

In 1-dimension, this theorem is illustrated by Figure 6.4. Since $\boldsymbol{f}$ is $K-$Lipschitz, the slope of the trajectory of the system is bounded between two times t_k. Since the trajectory contracts at discrete times, the trajectory is forced to converge toward the origin.

In the Theorem 6.3, the discretised system can be verified exponentially stable in $\mathscr{E}$ using the method from Chapter 5. Since the analytical expression of the discrete mapping $\boldsymbol{\phi}_t$ is unknown, the Jacobian of $\boldsymbol{\phi}_t$ must be evaluated with a guaranteed integration of the variational equation, as presented in Chapter 3.

Moreover, Theorem 6.3 proves that $\mathscr{E}$ is positive invariant with respect to the discretised system, but there is no guarantee that $\mathscr{E}$ is also positive invariant with the continuous system. This is why a different discretisation was used in Section 6.3.

6.5 Implementation

The Algorithm 3 is proposed to solve the problem introduced in Section 6.2, using the results from Section 6.3 and Section 6.4. This algorithm is the adaptation of Algorithm 2 for continuous systems. It is also computationally tractable. In this algorithm, two ellipsoidal propagation are made, one for each discretisation of the continuous system. As Algorithm 2, this algorithm has polynomial complexity.

Discussion

There is no automatic selection for the discretisation time δ. Several attempts may be required to tune δ.

With a small values of δ, the mappings $\boldsymbol{h}_{\text{Euler}}$ and $\boldsymbol{\phi}_\delta$ are similar, and so are $\mathcal{P}_{h_{\text{Euler}}}(\mathcal{E}(\boldsymbol{\Gamma}))$ and $\mathcal{P}_{\boldsymbol{\phi}_\delta}(\mathcal{E}(\boldsymbol{\Gamma}))$. But if δ is too small, then the pessimism in the computation of the ellipsoid compensates for the contraction of the mappings. So the algorithm is not able to conclude.

With a high values of δ, the ellipsoid $\mathcal{E}(\boldsymbol{\Gamma})$ contracts significantly. However, the pessimism in the computation of the Jacobian matrix of $\boldsymbol{\phi}_\delta$ increases. Moreover, if δ

Algorithm 3

```
Inputs $\boldsymbol{f}$,$\alpha_{\max}$,$\delta$
Outputs res, $\boldsymbol{\Gamma}$
$\boldsymbol{A} = \frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{x}}(\boldsymbol{0})$
$\boldsymbol{J} = \boldsymbol{I}_n + \delta \cdot \boldsymbol{A}$
$\boldsymbol{P}$ = solve_discrete_Lyapunov($\boldsymbol{J}$)
$\alpha = 1$
res = False
while $\alpha < \alpha_{\max}$ and res = False:
    $\boldsymbol{\Gamma} = 10^{-\alpha} \cdot \boldsymbol{P}^{-\frac{1}{2}}$
    $\boldsymbol{\Gamma}_{\text{Euler}} = \mathcal{E}^{-1}\left(\mathcal{P}_{h_{\text{Euler}}}\left(\mathcal{E}\left(\boldsymbol{\Gamma}\right)\right)\right)$ // with $\boldsymbol{h}_{\text{Euler}} : \boldsymbol{x} \to \boldsymbol{x} + \delta \cdot \boldsymbol{f}(\boldsymbol{x})$
    $\boldsymbol{\Gamma}_{\phi} = \mathcal{E}^{-1}\left(\mathcal{P}_{\boldsymbol{\phi}_{\delta}}\left(\mathcal{E}\left(\boldsymbol{\Gamma}\right)\right)\right)$
    res = is_definite_positive$\left(\boldsymbol{\Gamma}_{\text{Euler}}^{-2} - \boldsymbol{\Gamma}^{-2}\right)$ and is_definite_positive$\left(\boldsymbol{\Gamma}_{\phi}^{-2} - \boldsymbol{\Gamma}^{-2}\right)$
    $\alpha = \alpha + 1$
```

Computation of the positive invariant ellipsoid $\mathcal{E}(\boldsymbol{\Gamma})$ in which the continuous system is exponentially stable

is too big, then the discrete system $\boldsymbol{x}_{k+1} = \boldsymbol{h}_{\text{Euler}}(\boldsymbol{x}_k)$ becomes unstable.

When the system is barely stable, it can be challenging to find an ellipsoid that is positive invariant with respect to the two discrete systems. In this case, one can use different time steps or different ellipsoid candidate for the two discretised systems, thus separating the two problems.

6.6 Application

In this section, the stability analysis of a continuous system is illustrated with a continuous version of the application example of Section 5.6 from Chapter 5. To recall, two ROVs are controlled with a virtual structure approach to reach an equilateral triangular formation. While this system was represented as a discrete-time system in Section 5.6, it is now modelled as a continuous-time system.

6.6.1 Mathematical description of the system

As illustrated by Figure 6.5 the ROVs Inky and Blinky are described using polar coordinates. The position and speed of the robots are continuous-time variables. the polar horizontal coordinates of the ROVs are written

$$(d_b, \phi_b) \in \mathbb{R}^2 \text{(for Inky)},$$
$$(d_r, \phi_r) \in \mathbb{R}^2 \text{(for Blinky)}.$$

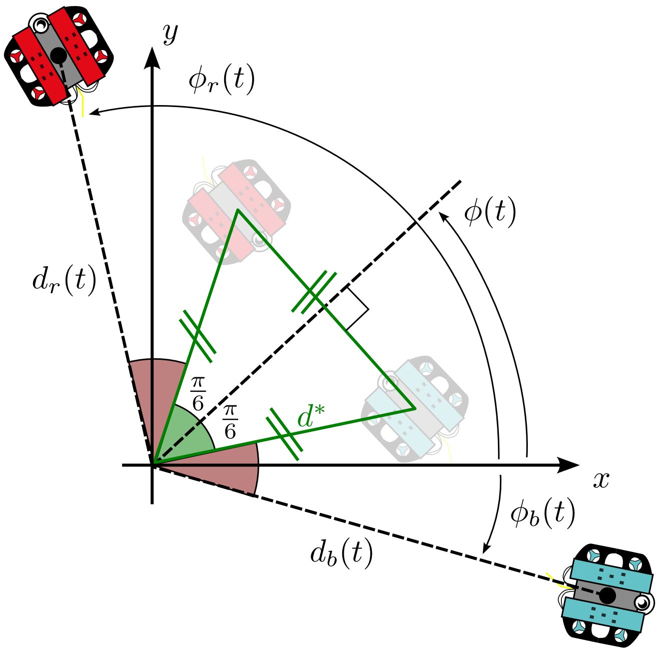


Figure 6.5: Definition of the continuous desired formation

The motion robots is described by the following ODE

$$\begin{cases} \ddot{d}_b(t) & = u_1(t), \\ \ddot{d}_r(t) & = u_2(t), \\ \ddot{\phi}_b(t) & = \frac{u_3(t)}{d_b(t)}, \\ \ddot{\phi}_r(t) & = \frac{u_4(t)}{d_r(t)}, \end{cases} \tag{6.23}$$

where $(u_1, u_2, u_3, u_4) \in \mathbb{R}^4$ are the continuous-time acceleration inputs of the system. As in Section 5.6, there is a singularity when $d_b(t) = 0$ or $d_r(t) = 0$ which will be avoided in the stability analysis. Then, as illustrated by Figure 6.5, the desired positions of the robot are now

$$\left(d^*, \phi(t) - \frac{\pi}{6}\right)^T \text{(for Inky)}, \tag{6.24}$$

$$\left(d^*, \phi(t) + \frac{\pi}{6}\right)^T \text{(for Blinky)}, \tag{6.25}$$

with the desired distance $d^* > 0$ and where the orientation of the triangle $\phi(t)$ is given by

$$\phi(t) = \frac{\phi_b(t) + \phi_r(t)}{2}. \tag{6.26}$$

The robots track their desired position with the following saturated proportional derivative controller

$$\begin{aligned}
u_1(t) &= s \cdot \arctan\left(k_{p,d} \cdot (d^* - d_b(t)) - k_{d,d} \cdot \dot{d}_b(t)\right), \\
u_2(t) &= s \cdot \arctan\left(k_{p,d} \cdot (d^* - d_r(t)) - k_{d,d} \cdot \dot{d}_r(t)\right), \\
u_3(t) &= s \cdot \arctan\left(k_{p,\phi} \cdot \left(\phi(t) - \frac{\pi}{6} - \phi_b(t)\right) - k_{d,\phi} \cdot \dot{\phi}_b(t)\right), \\
u_4(t) &= s \cdot \arctan\left(k_{p,\phi} \cdot \left(\phi(t) + \frac{\pi}{6} - \phi_r(t)\right) - k_{d,\phi} \cdot \dot{\phi}_r(t)\right),
\end{aligned} \tag{6.27}$$

which is the continuous counterpart of the controller in Section 5.6, with the saturation amplitude $s > 0$ and the controller gains $(k_{p,d}, k_{p,\phi}, k_{d,d}, k_{d,\phi}) > 0$. Finally, to describe the full system, consider the global continuous-time state vector $\boldsymbol{x} = (x_i)_{i \in [\![1:6]\!]} \in \mathbb{R}^6$ with

$$\begin{aligned}
x_1 &= d_b - d^* \\
x_2 &= d_r - d^* \\
x_3 &= \phi_r - \phi_b - \frac{\pi}{3} \\
x_4 &= \dot{d}_b \\
x_5 &= \dot{d}_r \\
x_6 &= \dot{\phi}_b \\
x_7 &= \dot{\phi}_r
\end{aligned} \tag{6.28}$$

The evolution of the global state vector is given by the ODE

$$\begin{aligned}
\dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t)), \\
\mathbf{0} &= \boldsymbol{f}(\mathbf{0}),
\end{aligned} \tag{6.29}$$

with

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} x_4 \\ x_5 \\ (x_7 - x_6) \\ s \cdot \arctan(-k_{p,d} \cdot x_1 - k_{d,d} \cdot x_4) \\ s \cdot \arctan(-k_{p,d} \cdot x_2 - k_{d,d} \cdot x_5) \\ \frac{s}{x_1 + d^*} \arctan\left(\frac{k_{p,\phi}}{2} x_3 - k_{d,\phi} \cdot x_6\right) \\ \frac{s}{x_2 + d^*} \arctan\left(-\frac{k_{p,\phi}}{2} x_3 - k_{d,\phi} \cdot x_7\right) \end{bmatrix}. \tag{6.30}$$

6.6.2 Finding a positive invariant ellipsoid with the Nagumo Theorem

The parameter values are the same as in Chapter 5, given by table 5.1. To find a PI ellipsoid, the system (6.29) is discretised with a time step $\delta > 0$ to obtain the following discrete-time system

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \boldsymbol{h}\left(\boldsymbol{x}_k\right), \\ &= \boldsymbol{x}_k + \delta \cdot \boldsymbol{f}\left(\boldsymbol{x}_k\right), \end{aligned} \tag{6.31}$$

with $\delta = 0.1$s, which is the exact system that was studied in Section 5.6. So a positive invariant ellipsoid $\mathscr{E}$ with respect to the discretised system is already computed. Thus, according to Theorem 6.2, the same ellipsoid $\mathscr{E}$ is also positive invariant with respect to the continuous-time system (6.29).

6.6.3 Proving the local exponential stability

To prove the local exponential stability, consider the following discrete system

$$\boldsymbol{x}_{k+1} = \boldsymbol{\phi}_\delta\left(\boldsymbol{x}_k\right) \tag{6.32}$$

where $\boldsymbol{\phi}_\delta$ is the flow function of the dynamical system (6.29) for a time $\delta = 0.1\,$s.

Compared to the Euler discretisation (6.31), there is no approximation in (6.32). However, the analytical expression of $\boldsymbol{\phi}_t$ is unknown. So the enclosure of the Jacobian of $\boldsymbol{\phi}_\delta$ must be computed by solving the variational equation with guaranteed integration, as presented in Chapter 1.

Nevertheless, since the time step is small, the two discrete systems (6.31) and (6.32) are close. So the ellipsoid $\mathscr{E}$ that is positive invariant with respect to the discrete system (6.31) will likely be positive invariant with respect to the exact discrete system (6.32).

Thus, with the Algorithm 2, the ellipsoid $\mathcal{P}_{\boldsymbol{\phi}_\delta}\left(\mathscr{E}\right)$ is computed and the inclusion $\mathcal{P}_{\phi_t}\left(\mathscr{E}\right) \subset \mathscr{E}$ is verified, as illustrated by Figure 6.6. Note that the Figure 6.6 is visually similar to Figure 5.10 that shows $\mathcal{P}_h\left(\mathscr{E}\right) \subset \mathscr{E}$.

As a result, $\mathscr{E}$ is positive invariant with respect to the discrete system (6.32). Therefore, using Theorem 6.3, the continuous system is exponentially stable in $\mathscr{E}$.

6.7 Conclusion

This chapter presents a numerical method to compute a positive invariant ellipsoid with respect to a high-dimensional nonlinear continuous-time dynamical system such that the system is exponentially stable in the ellipsoid. This method is an adaptation of the method from Chapter 5 for continuous systems. To our knowledge, it is the first guaranteed numerical method that can compute an ellipsoidal domain of attraction for a high-dimensional continuous-time nonlinear system. For continuous systems, the proof of positive invariance and exponential stability are separated via the study of two different discretised systems.

Moreover, the method assumes that the mapping of the systems is non-singular, which is the case for continuous-time systems most of the time. However, Chapter 8, which adapts the method to synchronous hybrid systems, deals with singular mappings. Thus, the adaptation of the method to singular mappings is first discussed in Chapter 7.

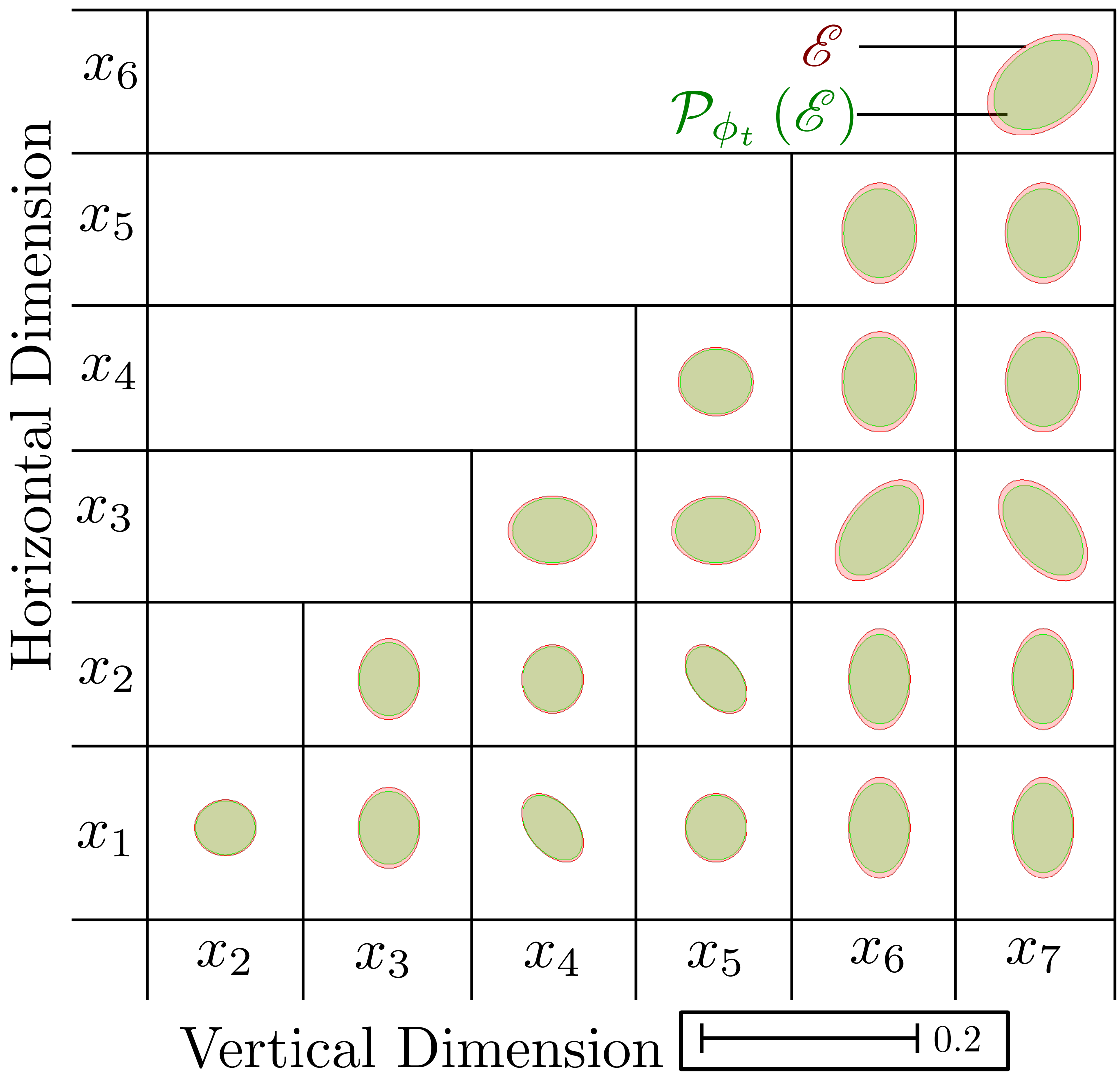


Figure 6.6: Representation of the 7-dimensional ellipsoids $\mathscr{E}$ (red) and $\mathcal{P}_{\phi_t}(\mathscr{E})$ (green) with orthogonal projections on the 2-dimensional planes $(0, x_i, x_j)$ with $i < j$.

The results of this chapter have been published in the Automatica journal [62]. In the following chapters, this method will be adapted to study hybrid nonlinear systems.

Chapter 7

Propagation of Degenerate ellipsoids

7.1 Introduction

Chapter 6 presented a new numerical method to compute positive invariant ellipsoid for continuous-time systems. To study the formation control of a group of robots, this method must be adapted to synchronous hybrid systems, which is the topic of Chapter 8.

The current chapter tackles an issue that must be solved before adapting the stability analysis to synchronous hybrid systems. The numerical method of Chapter 6 uses the guaranteed ellipsoidal propagation method presented in Chapter 4. This method can propagate an ellipsoid with a non-signular mapping. However, with hybrid systems, the ellipsoids are often propagated with singular mappings. With a singular mapping, an ellipsoid can be flattened, making it a degenerate ellipsoid. So, this chapter extends the guaranteed ellipsoidal propagation to the singular case and the degenerate ellipsoids. The method is extended using an eigenvalue decomposition of the shape matrix and the addition of new non-zero eigenvalues when necessary.

The chapter is organised as follows. Section 7.2 presents the specificity of degenerate ellipsoids. Section 7.3 introduces the generalisation of the guaranteed ellipsoidal propagation. Section 7.4 presents the algorithmic implementation of the generalised method. Finally, Section 7.5 shows some results obtained with singular mappings.

7.1.1 Singular mapping in hybrid system

Hybrid systems can have singular mappings with discrete-time measurements, event triggering or shock effects. A common example of singular mapping is the isobath rebound. An isobath is a contour line of constant depth. As illustrated by Figure 7.1, consider an underwater robot moving in a straight line. This robot can measure its distance to the seabed, so it can detect an isobath crossing. Assume that the robot must turn back when it reaches the isobath defined by $x_2 = \sin(x_1)$, as illustrated in the figure. Depending on its initial position, the robot does not reach the isobath at the same time, or at the same position. Given an ellipsoid of initial positions $\mathscr{E} \subset \mathbb{R}^- \times [-1, 1]$ where does the robot cross the isobath?

To enclose the crossing positions, one can propagate the initial ellipsoid with the mapping

$$\begin{aligned} \boldsymbol{g} : \mathbb{R}^- \times [-1,1] &\to \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \times [-1,1] \\ \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} &\mapsto \begin{bmatrix} \arcsin(x_2) \\ x_2 \end{bmatrix}, \end{aligned} \tag{7.1}$$

which is a projection on the isobath. However, this mapping is singular because its Jacobian matrix, given by

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}(\boldsymbol{x}) = \begin{bmatrix} 0 & \frac{1}{\sqrt{1-x_2^2}} \\ 0 & 1 \end{bmatrix}, \tag{7.2}$$

has a null eigenvalues. So the guaranteed propagation method cannot be used with the mapping $\boldsymbol{h}$. However, the crossing positions could be enclosed by a ellipsoid like $\mathscr{E}_{\text{out}}$. In this context, this chapter proposes to extend the guaranteed propagation of ellipsoids to singular mappings like $\boldsymbol{g}$ to compute an enclosing ellipsoid like $\mathscr{E}_{\text{out}}$.

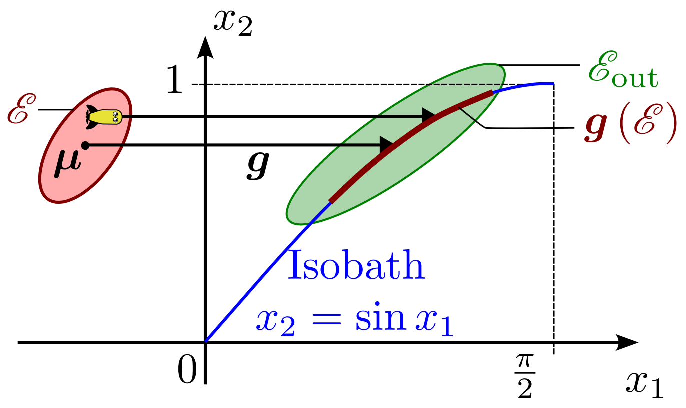
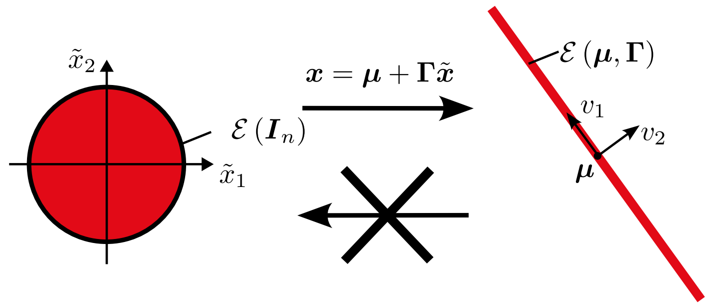


Figure 7.1: Isobath crossing. The projection mapping $\boldsymbol{g}$ is singular. How to compute an enclosing ellipsoid $\mathscr{E}_{\text{out}}$ such that $\boldsymbol{g}(\mathscr{E}) \subseteq \mathscr{E}_{\text{out}}$?

7.2 Degenerate ellipsoids

Degenerate ellipsoids are ellipsoids with a non-invertible shape matrix, such that the ellipsoids are flat in some directions, as presented in [4] and illustrated by Figure 7.2. Unlike non-degenerate ellipsoids, they can't be described by a quadratic form. However, they can be described by an affine form, with a positive semi-definite symmetric matrix. Definition 7.1 proposes a general definition of ellipsoids to include degenerate ellipsoids. With this definition, the ellipsoid is degenerate when $\boldsymbol{\Gamma}$ is non-invertible and non-degenerate when $\boldsymbol{\Gamma}$ is definite positive.



$\tilde{x}_2$
$\tilde{x}_1$
$\mathcal{E}(\boldsymbol{I}_n)$
$\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{\Gamma}\tilde{\boldsymbol{x}}$
$\mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma})$
v_1
v_2
$\boldsymbol{\mu}$

Figure 7.2: Illustration of a 2 dimensional degenerate ellipsoid. The ellipsoid is flat in the direction of the eigenvector v_2.

Definition 7.1. (General definition of ellipsoids) An ellipsoid is a subset of $\mathbb{R}^n$ described by a unique midpoint $\boldsymbol{\mu} \in \mathbb{R}^n$ and a unique symmetric positive semi-definite matrix $\boldsymbol{\Gamma} \in S_n$ such that

$$\mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma}) := \boldsymbol{\mu} + \boldsymbol{\Gamma} \cdot \mathcal{E}(\boldsymbol{I}_n) \tag{7.3}$$

The same notation will be used for non-degenerate and degenerate ellipsoids, namely $\mathscr{E}$, $\mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma})$ and $\mathcal{E}(\boldsymbol{\Gamma})$. When necessary the ellipsoid will be specified degenerate or not. A degenerate ellipsoid $\mathscr{E}$ is flat, so it has an empty interior and it is equal to its border $\partial\mathscr{E} = \mathscr{E}$.

Some properties hold for degenerate ellipsoids. For all $\boldsymbol{\Gamma} \in S_n$, all non-null real scalar $\alpha \in \mathbb{R}^*$, and all invertible matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, one has

$$\alpha \cdot \mathcal{E}(\boldsymbol{\Gamma}) = \mathcal{E}(\alpha \cdot \boldsymbol{\Gamma}), \tag{7.4}$$

$$\boldsymbol{A} \cdot \mathcal{E}(\boldsymbol{\Gamma}) = \mathcal{E}\left(\left(\boldsymbol{A}\boldsymbol{\Gamma}^2\boldsymbol{A}^T\right)^{\frac{1}{2}}\right). \tag{7.5}$$

The semi-axis of degenerate ellipsoids is also described by the eigenvalues and the eigenvectors of $\boldsymbol{\Gamma}$. The eigenvectors associated with null eigenvalues indicate the direction where the ellipsoid is flat, as illustrated by Figure 7.2.

Eigenvalue decomposition The eigenvalues decomposition of $\boldsymbol{\Gamma} \in S_n$, used in this chapter, is given by

$$\boldsymbol{\Gamma} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T \tag{7.6}$$

where $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ is the orthonormal matrix whose columns are the eigenvectors of $\boldsymbol{\Gamma}$ and where $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ is the diagonal matrix whose diagonal elements are the eigenvalues.

Whereas $\boldsymbol{\Gamma}$ has no inverse when the ellipsoid is degenerate, it has a pseudo-inverse $\boldsymbol{\Gamma}^+$ equal to

$$\boldsymbol{\Gamma}^+ = \boldsymbol{U}\boldsymbol{D}^+\boldsymbol{U}^T \tag{7.7}$$

where $\boldsymbol{D}^+$, the pseudo inverse of $\boldsymbol{D}$ is formed by replacing every non-zero diagonal element with its inverse.

Inclusion with degenerate ellipsoid The inclusion of degenerate ellipsoids cannot be tested with Theorem 4.5 because it inverses the shape matrices. However, for two ellipsoids $\mathscr{E}_1$ and $\mathscr{E}_2$, if $\mathscr{E}_2$ is non-degenerate, the inclusions $\mathscr{E}_1 \subseteq \mathscr{E}_2$ and $\mathscr{E}_1 \subset \mathscr{E}_2$ can be tested with Theorem 7.1. If $\mathscr{E}_2$ is degenerate, the strict inclusion is not relevant because $\mathscr{E}_2$ has an empty interior. Moreover, when $\mathscr{E}_2$ is degenerate the inclusion $\mathscr{E}_1 \subseteq \mathscr{E}_2$ is hard to verify numerically because one has to prove that $\mathscr{E}_1$ is flat in the same directions as $\mathscr{E}_2$.

Theorem 7.1. *Let $\mathscr{E}_1$ be an ellipsoid of $\mathbb{R}^n$ and let $\mathscr{E}_2$ be a non-degenerate ellipsoid of $\mathbb{R}^n$ such that $\mathscr{E}_1$ and $\mathscr{E}_2$ have the same midpoint $\boldsymbol{\mu} \in \mathbb{R}^n$. Consider the shapes matrices $\boldsymbol{\Gamma}_1 \in S_n$ and $\boldsymbol{\Gamma}_2 \in S_n^+$ such that $\mathscr{E}_1 = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma}_1)$ and $\mathscr{E}_2 = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma}_2)$. Then, one has*

$$(\mathscr{E}_1 \subseteq \mathscr{E}_2) \Leftrightarrow \left(\boldsymbol{I}_n - \boldsymbol{\Gamma}_1 \boldsymbol{\Gamma}_2^{-2} \boldsymbol{\Gamma}_1 \succeq 0\right), \tag{7.8}$$

$$(\mathscr{E}_1 \subset \mathscr{E}_2) \Leftrightarrow \left(\boldsymbol{I}_n - \boldsymbol{\Gamma}_1 \boldsymbol{\Gamma}_2^{-2} \boldsymbol{\Gamma}_1 \succ 0\right). \tag{7.9}$$

Proof. Sketch of a proof. Let $\boldsymbol{e} \in \mathbb{R}^n$ such that $\|\boldsymbol{e}\|_2 = 1$. Let $\boldsymbol{x} = \boldsymbol{\Gamma}_1 \cdot \boldsymbol{e} \in \partial\mathscr{E}_1$. The key idea is that

$$\begin{aligned} 1 - \|\boldsymbol{x}\|_{\boldsymbol{\Gamma}_2^{-2}} &= 1 - \|\boldsymbol{\Gamma}_1 \cdot \boldsymbol{e}\|_{\boldsymbol{\Gamma}_2^{-2}} \\ &= \boldsymbol{e}^T \boldsymbol{e} - \boldsymbol{e}^T \left(\boldsymbol{\Gamma}_1 \boldsymbol{\Gamma}_2^{-2} \boldsymbol{\Gamma}_1\right) \boldsymbol{e}, \\ &= \boldsymbol{e}^T \left(\boldsymbol{I}_n - \boldsymbol{\Gamma}_1 \boldsymbol{\Gamma}_2^{-2} \boldsymbol{\Gamma}_1\right) \boldsymbol{e}, \end{aligned} \tag{7.10}$$

and

$$\boldsymbol{x} \in \mathscr{E}_2 \Leftrightarrow \|\boldsymbol{x}\|_{\boldsymbol{\Gamma}_2^{-2}} \leq 1, \tag{7.11}$$

from which (7.8) and (7.9) are deduced. □

7.3 General propagation method

This section presents a generalisation of the guaranteed ellipsoidal propagation method presented in Section 4.5 from Chapter 3, to apply it to degenerate ellipsoid and singular mappings. As in Chapter 3, consider the following differentiable nonlinear mapping

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}), \boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^n \tag{7.12}$$

and the ellipsoid $\mathscr{E}$ of $\mathbb{R}^n$. The image set

$$\boldsymbol{h}(\mathscr{E}) = \{\boldsymbol{y} \in \mathbb{R}^n | \exists \boldsymbol{x} \in \mathscr{E}, \boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x})\}, \tag{7.13}$$

must be evaluated. Compared to Chapter 3 the ellipsoid $\mathscr{E}$ is not assumed non-degenerate and $\boldsymbol{h}$ is not assumed non-singular. To evaluate $\boldsymbol{h}(\mathscr{E})$, and outer enclosing ellipsoid $\mathscr{E}_{\text{out}}$ of $\mathbb{R}^n$ is computed such that

$$\boldsymbol{h}(\mathscr{E}) \subseteq \mathscr{E}_{\text{out}}. \tag{7.14}$$

The ellipsoid $\mathscr{E}_{\text{out}}$ can also be degenerate. The ellipsoid $\mathscr{E}_{\text{out}}$ is computed with Theorem 7.2, the generalisation of Theorem 4.5. To introduce this new theorem, some preliminary ideas are given in Section 7.3.1. The theorem is then presented in Section 7.3.2.

7.3.1 Preliminary ideas

To recall, the propagation method presented in Chapter 3 consists in inflating the ellipsoid obtained by linearising $\boldsymbol{h}$, as illustrated by Figure 7.3. Given the initial ellipsoid $\mathscr{E} = \mathcal{E}\left(\boldsymbol{\mu}, \boldsymbol{\Gamma}\right)$, and the Jacobian matrix

$$\boldsymbol{J} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\left(\boldsymbol{\mu}\right),$$

one can obtain the ellipsoid $\mathscr{E}_l = \mathcal{E}\left(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{\Gamma}_l\right)$ with

$$\boldsymbol{\mu}_{\text{out}} = \boldsymbol{h}\left(\boldsymbol{\mu}\right), \tag{7.15}$$

$$\boldsymbol{\Gamma}_l = \left(\boldsymbol{J}\boldsymbol{\Gamma}^2\boldsymbol{J}^T\right)^{\frac{1}{2}}. \tag{7.16}$$

Then the original method is able to compute and inflation gain ρ to obtain the outer enclosure

$$\mathscr{E}_{\text{out}} = \mathcal{E}\left(\boldsymbol{\mu}_{\text{out}}, (1+\rho)\cdot\boldsymbol{\Gamma}_l\right). \tag{7.17}$$

However, in the singular case, $\mathscr{E}$ is degenerate or $\boldsymbol{h}$ is singular. So $\mathscr{E}_l$ is degenerate as illustrated by Figure 7.4. In other words, $\boldsymbol{\Gamma}_l$ is non-invertible because either $\boldsymbol{J}$ and/or $\boldsymbol{\Gamma}$ are non-invertible. The fact that $\mathscr{E}_l$ is degenerate creates two problems:

1. One cannot compute ρ with the original formula which contains the inverse of $\boldsymbol{\Gamma}_l$
2. While $\mathscr{E}_l$ is flat in some direction, $\boldsymbol{h}\left(\mathscr{E}\right)$ may be not flat in the same direction. Thus, even an infinite inflation of $\mathscr{E}_l$ cannot enclose $\boldsymbol{h}\left(\mathscr{E}\right)$.

To solve the first problem, the formula of ρ must be updated using pseudo inverse. To solve the second problem, the flatness of $\mathscr{E}_l$ must be removed to obtain an intermediate ellipsoid $\mathscr{E}_m$ which will be inflated to compute $\mathscr{E}_{\text{out}}$.

7.3.1.1 Computation of $\mathscr{E}_m$

To avoid pessimism in the method, the shape of $\mathscr{E}_m$ must be close to $\boldsymbol{f}\left(\mathscr{E}\right)$. Consider the eigenvalue decomposition of $\boldsymbol{\Gamma}_l$, given by

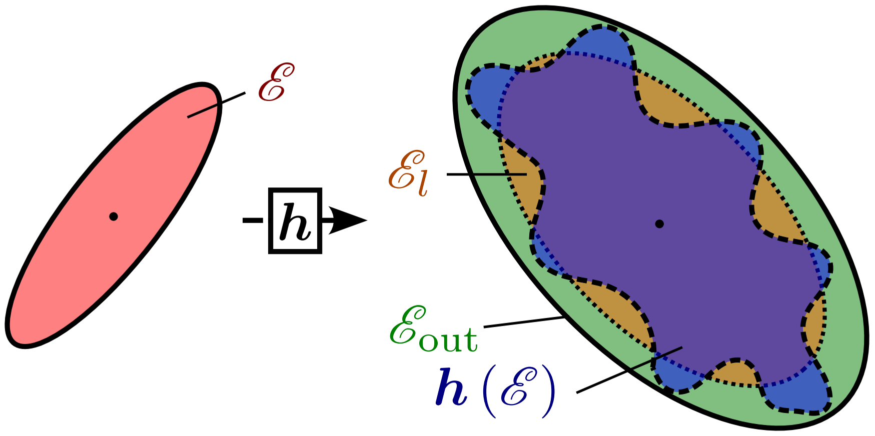
$$\boldsymbol{\Gamma}_l = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T \tag{7.18}$$

with the orthonormal matrix $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ and the diagonal matrix $\boldsymbol{D} \in \mathbb{R}^{n\times n}$. A choice for $\mathscr{E}_m$ can be

$$\mathscr{E}_m = \mathcal{E}\left(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{M}\right) \tag{7.19}$$

with the shape matrix

$$\boldsymbol{M} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^T \tag{7.20}$$



$\mathscr{E}$
$\boldsymbol{h}$
$\mathscr{E}_l$
$\mathscr{E}_{\text{out}}$
$\boldsymbol{h}\left(\mathscr{E}\right)$

Figure 7.3: Non-singular guaranteed ellipsoidal propagation. $\mathscr{E}_{\text{out}}$ is obtained by inflating $\mathscr{E}_l$.

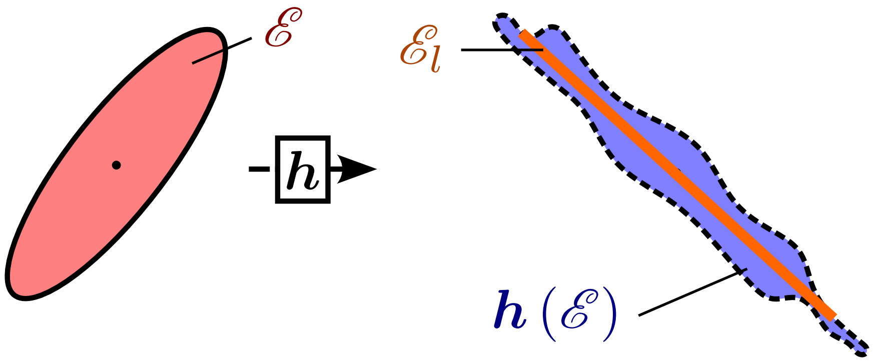
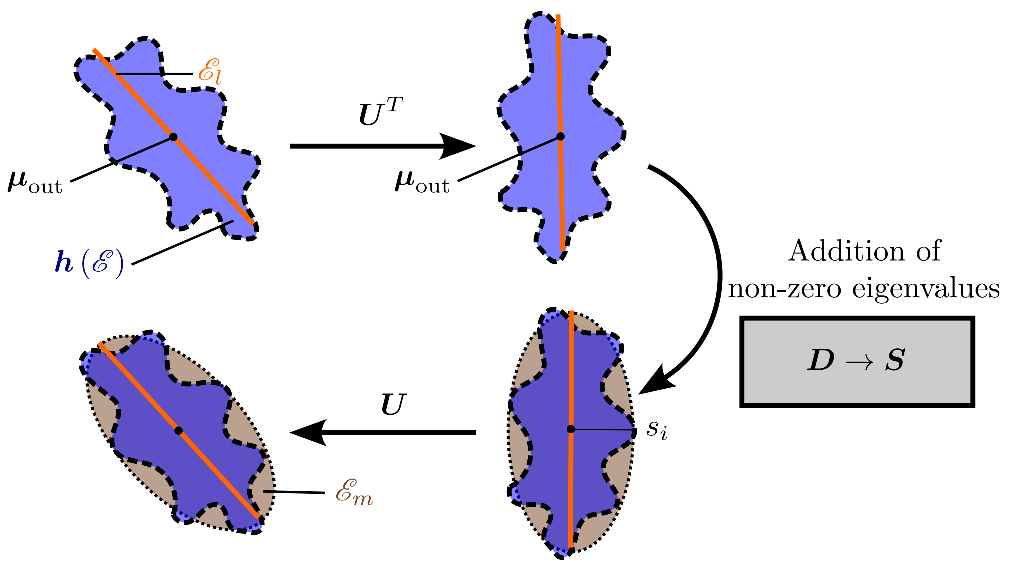


Figure 7.4: Singular ellipsoidal propagation. An inflation of $\mathscr{E}_l$ cannot enclose $\boldsymbol{h}\left(\mathscr{E}\right)$



$\mathscr{E}_l$
$\boldsymbol{U}^T$
$\boldsymbol{\mu}_{\text{out}}$
$\boldsymbol{\mu}_{\text{out}}$
$\boldsymbol{h}(\mathscr{E})$
Addition of non-zero eigenvalues
$\boldsymbol{D} \to \boldsymbol{S}$
$\boldsymbol{U}$
s_i
$\mathscr{E}_m$

Figure 7.5: The ellipsoid $\mathscr{E}_m$ is obtained by adding non-zero singular values.

and the diagonal matrix

$$\boldsymbol{S} = \operatorname{diag}(s_1, s_2, \ldots, s_n),$$
$$s_i = \begin{cases} d_i & \text{if } \sigma_i > 0, \\ q_i \cdot \max_{\boldsymbol{x} \in \mathscr{E}} \left| \boldsymbol{e}_i^T \boldsymbol{U}^T (\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{\mu}_{\text{out}}) \right| & \text{else,} \end{cases} \tag{7.21}$$

where the $q_i > 0$ are design parameters and where d_i is the i^{th} diagonal element of $\boldsymbol{D}$. The matrix $\boldsymbol{M}$ is obtained by replacing the zero-value eigenvalues of $\boldsymbol{\Gamma}_l$ by the term

$$s_i = q_i \cdot \max_{\boldsymbol{x} \in \mathscr{E}} \left| \boldsymbol{e}_i^T \boldsymbol{U}^T (\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{\mu}_{\text{out}}) \right| \tag{7.22}$$

which makes $\mathscr{E}_m$ have a similar size than $\boldsymbol{f}(\mathscr{E})$ in the direction where $\mathscr{E}_l$ is flat, as illustrated by Figure 7.5. The parameters q_i have been added empirically to give less pessimism depending on the shape of $\boldsymbol{f}(\mathscr{E})$. Note that $\mathscr{E}_m$ is only flat in the directions where $\boldsymbol{f}(\mathscr{E})$ is flat.

7.3.1.2 Computation of ρ

Then, the inflation gain ρ can be computed as

$$\rho = \min \left\{ \rho \in \mathbb{R}^+ | \forall \widetilde{\boldsymbol{x}} \in \mathcal{E}(\boldsymbol{I}_n), \left\| \widetilde{\boldsymbol{b}}(\widetilde{\boldsymbol{x}}) \right\|_2 \leq \rho \right\},$$

with the function

$$\begin{aligned} \widetilde{\boldsymbol{b}} :& \mathbb{R}^n \mapsto \mathbb{R}^n, \\ & \widetilde{\boldsymbol{x}} \to \boldsymbol{M}^+ \cdot (\boldsymbol{f}(\boldsymbol{\Gamma} \cdot \widetilde{\boldsymbol{x}} + \boldsymbol{\mu}) - \boldsymbol{\mu}_{\text{out}}) - \widetilde{\boldsymbol{x}}. \end{aligned} \tag{7.23}$$

The function $\widetilde{\boldsymbol{b}}$ is similar to the original one, but the inverse of $\boldsymbol{\Gamma}_l$ has been replace by the pseudo inverse of $\boldsymbol{M}$.

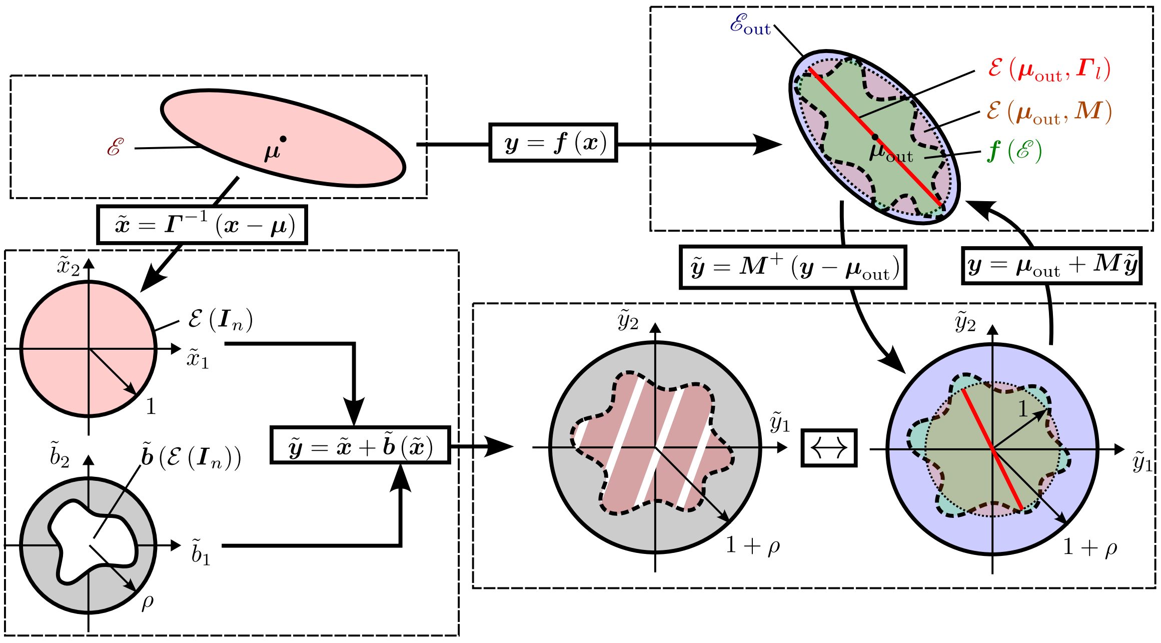

$\mathscr{E}$
$\boldsymbol{\mu}$
$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$
$\mathscr{E}_{\text{out}}$
$\mathcal{E}(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{\Gamma}_l)$
$\mathcal{E}(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{M})$
$\boldsymbol{f}(\mathscr{E})$
$\boldsymbol{\mu}_{\text{out}}$
$\tilde{\boldsymbol{x}} = \boldsymbol{\Gamma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})$
$\tilde{\boldsymbol{y}} = \boldsymbol{M}^{+}(\boldsymbol{y} - \boldsymbol{\mu}_{\text{out}})$
$\boldsymbol{y} = \boldsymbol{\mu}_{\text{out}} + \boldsymbol{M}\tilde{\boldsymbol{y}}$
$\tilde{x}_2$
$\tilde{x}_1$
$\mathcal{E}(\boldsymbol{I}_n)$
1
$\tilde{b}_2$
$\tilde{b}_1$
$\tilde{\boldsymbol{b}}(\mathcal{E}(\boldsymbol{I}_n))$
ρ
$\tilde{\boldsymbol{y}} = \tilde{\boldsymbol{x}} + \tilde{\boldsymbol{b}}(\tilde{\boldsymbol{x}})$
$\tilde{y}_2$
$\tilde{y}_1$
$1+\rho$
$\leftrightarrow$
$\tilde{y}_2$
$\tilde{y}_1$
1
$1+\rho$

Figure 7.6: Illustration of Theorem 7.2 in the singular case

7.3.2 Presentation of the Theorem

This section present the Theorem 7.2 which is illustrated by Figure 7.6. Note that, in a non-singular case, the ellipsoids $\mathcal{E}(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{\Gamma}_l)$ and $\mathcal{E}(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{M})$ are identical.

Theorem 7.2. *Let $\mathscr{E} = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\Gamma})$ be an ellipsoid of $\mathbb{R}^n$ with $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\boldsymbol{\Gamma} \in \mathcal{S}_n$. Let $\boldsymbol{f}$ be a $\mathcal{C}^1$ mapping over $\mathbb{R}^n$. It's Jacobian matrix at the origin is written,*

$$\boldsymbol{J} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}(\boldsymbol{\mu}). \tag{7.24}$$

Let

$$\begin{aligned}\boldsymbol{\Gamma}_l &= \left(\boldsymbol{J}\boldsymbol{\Gamma}^2\boldsymbol{J}^T\right)^{\frac{1}{2}},\\ &\in \mathcal{S}_n,\end{aligned}$$

and consider the eigenvalue decomposition of $\boldsymbol{\Gamma}_l$, given by

$$\boldsymbol{\Gamma}_l = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T \tag{7.25}$$

with the orthonormal matrix $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ and the diagonal matrix $\boldsymbol{D} \in \mathbb{R}^{n\times n}$. Consider the point

$$\boldsymbol{\mu}_{out} = \boldsymbol{f}(\boldsymbol{\mu}), \tag{7.26}$$

and the matrix

$$\boldsymbol{\Gamma}_{out} = (1+\rho)\cdot\boldsymbol{M}, \tag{7.27}$$

with the inflation gain

$$\rho = \min\left\{\rho \in \mathbb{R}^+ | \forall \widetilde{\boldsymbol{x}} \in \mathcal{E}\left(\boldsymbol{I}_n\right), \left\|\widetilde{\boldsymbol{b}}\left(\widetilde{\boldsymbol{x}}\right)\right\|_2 \leq \rho\right\}, \tag{7.28}$$

where

$$\begin{aligned} \widetilde{\boldsymbol{b}} :& \mathbb{R}^n \mapsto \mathbb{R}^n, \\ & \widetilde{\boldsymbol{x}} \to \boldsymbol{M}^+ \cdot \left(\boldsymbol{f}\left(\boldsymbol{\Gamma} \cdot \widetilde{\boldsymbol{x}} + \boldsymbol{\mu}\right) - \boldsymbol{\mu}_{out}\right) - \widetilde{\boldsymbol{x}}. \end{aligned} \tag{7.29}$$

with $\boldsymbol{M} \in \mathbb{R}^{n \times n}$ and its pseudo-inverse $\boldsymbol{M}^+$ are given by
case 1 (General): if $\boldsymbol{\Gamma}_l$ is invertible, then

$$\boldsymbol{M} = \boldsymbol{\Gamma}_l, \tag{7.30}$$

$$\boldsymbol{M}^+ = \boldsymbol{\Gamma}_l^{-1}, \tag{7.31}$$

case 2 (Singularity): if $\boldsymbol{\Gamma}_l$ is non-invertible, then

$$\boldsymbol{M} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^T, \tag{7.32}$$

$$\boldsymbol{M}^+ = \boldsymbol{U}\boldsymbol{S}^+\boldsymbol{U}^T, \tag{7.33}$$

$$\boldsymbol{S} = diag\left(s_1, s_2, \ldots, s_n\right), \tag{7.34}$$

$$s_i = \begin{cases} d_i & if d_i > 0, \\ q_i \cdot \max\limits_{\boldsymbol{x} \in \mathscr{E}} \left|\boldsymbol{e}_i^T \boldsymbol{U}^T \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{out}\right)\right| & else, \end{cases} \tag{7.35}$$

where d_i is the i^{th} diagonal element of $\boldsymbol{D}$, $\boldsymbol{e}_i$ is the i^{th} unitary vector of the Cartesian base of $\mathbb{R}^n$ and $q_i > 0$ are design parameters. Then, the ellipsoid $\mathscr{E}_{out} = \mathcal{E}\left(\boldsymbol{\mu}_{out}, \boldsymbol{\Gamma}_{out}\right)$ is an outer enclosure of $\boldsymbol{f}\left(\mathscr{E}\right)$.

Proof. The case 1 refers to Theorem 4.5. To study the case 2, let $\boldsymbol{x} \in \mathscr{E}$. From (7.3), there exist $\widetilde{\boldsymbol{x}} \in \mathcal{E}\left(\boldsymbol{I}_n\right)$ such that

$$\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{\Gamma} \cdot \widetilde{\boldsymbol{x}}. \tag{7.36}$$

Then, let

$$\widetilde{\boldsymbol{y}} = \widetilde{\boldsymbol{x}} + \boldsymbol{b}\left(\widetilde{\boldsymbol{x}}\right) \tag{7.37}$$

From (7.29), one has

$$\begin{aligned} \boldsymbol{M} \cdot \widetilde{\boldsymbol{y}} &= \boldsymbol{M}\left(\boldsymbol{M}^+ \cdot \left(\boldsymbol{f}\left(\boldsymbol{\Gamma} \cdot \widetilde{\boldsymbol{x}} + \boldsymbol{\mu}\right) - \boldsymbol{\mu}_{\text{out}}\right)\right), \\ &\overset{(7.36)}{=} \boldsymbol{M}\boldsymbol{M}^+ \cdot \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right), \end{aligned} \tag{7.38}$$

Moreover, from Appendix 10, one gets

$$\boldsymbol{M}\boldsymbol{M}^+\left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right) = \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right). \tag{7.39}$$

Thus

$$\boldsymbol{M} \cdot \widetilde{\boldsymbol{y}} = \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right). \tag{7.40}$$

As a result,

$$\begin{aligned} \boldsymbol{f}\left(\boldsymbol{x}\right) &= \boldsymbol{\mu}_{\text{out}} + \boldsymbol{M} \cdot \widetilde{\boldsymbol{y}}, \\ &\overset{(7.27)}{=} \boldsymbol{\mu}_{\text{out}} + \boldsymbol{\Gamma}_{\text{out}} \left(\frac{1}{(1+\rho)} \widetilde{\boldsymbol{y}}\right). \end{aligned} \tag{7.41}$$

Then, since $\widetilde{\boldsymbol{x}} \in \mathcal{E}\left(\boldsymbol{I}_n\right)$, one has $\left\|\widetilde{\boldsymbol{x}}\right\|_2 \leq 1$. So, from (7.37), (7.28) and the triangle inequality, one deduces

$$\begin{aligned} \left\|\frac{1}{1+\rho}\widetilde{\boldsymbol{y}}\right\|_2 &= \frac{1}{1+\rho}\left\|\widetilde{\boldsymbol{y}}\right\|_2 \\ &\leq \frac{1}{1+\rho} \cdot \left(\left\|\widetilde{\boldsymbol{x}}\right\|_2 + \left\|\boldsymbol{b}\left(\widetilde{\boldsymbol{x}}\right)\right\|_2\right), \qquad (7.42) \\ &\leq \frac{1+\rho}{1+\rho}, \\ &\leq 1. \qquad (7.43) \end{aligned}$$

Finally, from (7.41) and (7.43), one has $\boldsymbol{f}\left(\boldsymbol{x}\right) \in \mathscr{E}_{\text{out}}$. Therefore,

$$\boldsymbol{f}\left(\mathscr{E}\right) \subseteq \mathscr{E}_{\text{out}}. \tag{7.44}$$

□

In the degenerate case, the propagation operator $\mathcal{P}_h$ can be given by

$$\mathcal{P}_h\left(\mathscr{E}\right) = \boldsymbol{h}\left(\boldsymbol{\mu}\right) + (1+\rho) \cdot \left(\boldsymbol{J} \cdot \left(\mathscr{E} - \boldsymbol{\mu}\right) \oplus \mathcal{E}\left(\boldsymbol{U}\left(\boldsymbol{S} - \boldsymbol{D}\right)\boldsymbol{U}^T\right)\right). \tag{7.45}$$

7.4 Implementation

Based on Theorem 7.2, The Algorithm 4 propose a generalisation of the Algorithm 1. is also defined for degenerate ellipsoid such that $\mathscr{E}_{\text{out}} = \mathcal{P}_h\left(\mathscr{E}\right)$.

In this algorithm, the centred form is used to compute the boxes $[\boldsymbol{s}]$ and $\left[\widetilde{\boldsymbol{b}}\right]$ such that

$$\boldsymbol{U}^T\left(\boldsymbol{f}\left(\mathscr{E}\right) - \boldsymbol{\mu}_{\text{out}}\right) \subseteq [\boldsymbol{s}] \tag{7.46}$$

$$\widetilde{\boldsymbol{b}}\left(\mathscr{E}\right) \subseteq \left[\widetilde{\boldsymbol{b}}\right] \tag{7.47}$$

Thus, the term $\max_{\boldsymbol{x} \in \mathscr{E}} \left|\boldsymbol{e}_i^T \boldsymbol{U}^T \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right)\right|$ can be overestimated by the upper bound of the i^{th} interval of $[\boldsymbol{s}]$, namely $\text{ub}\left([\boldsymbol{s}_i]\right)$. Moreover, as in the original method, the inflation gain ρ is overestimated by the upper bound of the norm of $\left[\widetilde{\boldsymbol{b}}\right]$, namely $\text{ub}\left(\left\|\left[\widetilde{\boldsymbol{b}}\right]\right\|\right)$

Algorithm 4 Outer ellipsoidal enclosures - with the degenerate case

```
Inputs $\mathscr{E}$,$\{q_1, q_2, \ldots, q_n\}$,$\{\alpha_1, \alpha_2\}$
Outputs $\mathcal{P}_h(\mathscr{E})$
 1: $(\boldsymbol{\mu}, \boldsymbol{\Gamma}) = \boldsymbol{\mathcal{E}}^{-1}(\mathscr{E})$
 2: $\boldsymbol{J} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\boldsymbol{\mu})$ // can be an approximation
 3: $[\boldsymbol{J}] = \text{EncloseJacobian}(\boldsymbol{h}, \mathscr{E})$ // such that $\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\mathscr{E}) \subseteq [\boldsymbol{J}]$
 4: $\boldsymbol{\Gamma}_l = \left(\boldsymbol{J}\boldsymbol{\Gamma}^2\boldsymbol{J}^T\right)^{\frac{1}{2}}$
 5: if $|\det(\boldsymbol{\Gamma}_l)| > \alpha_1$ then
 6:     $\boldsymbol{M} = \boldsymbol{\Gamma}_l$
 7: else
 8:     $(\boldsymbol{U}, \boldsymbol{D})$= diagonalization($\boldsymbol{\Gamma}_l$)
 9:     $[\boldsymbol{s}] = \boldsymbol{U}^T \cdot [\boldsymbol{J}] \cdot \boldsymbol{\Gamma} \cdot [1_n]$
10:     for $i$ from $1$ to $n$ do
11:         if $d_i > \alpha_2$ then
12:             $s_i = d_i$
13:         else
14:             $s_i = q_i \cdot \text{ub}([\boldsymbol{s}_i])$
15:         end if
16:     end for
17:     $\boldsymbol{S} = \text{diag}(s_1, s_2, \ldots, s_n)$
18:     $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^T$
19: end if
20: $\left[\widetilde{\boldsymbol{b}}\right] = \left(\boldsymbol{M}^+ \cdot [\boldsymbol{J}] \cdot \boldsymbol{\Gamma} - \boldsymbol{I}_n\right) \cdot [\mathbf{1}]_n$
21: $\rho = \text{ub}\left(\left\|\left[\widetilde{\boldsymbol{b}}\right]\right\|\right)$
22: $\boldsymbol{\mu}_{\text{out}} = \boldsymbol{h}(\boldsymbol{\mu})$
23: $\boldsymbol{\Gamma}_{\text{out}} = (1 + \rho) \cdot \boldsymbol{M}$
24: $\mathcal{P}_h(\mathscr{E}) = \boldsymbol{\mathcal{E}}(\boldsymbol{\mu}_{\text{out}}, \boldsymbol{\Gamma}_{\text{out}})$
```

The matrices $\boldsymbol{U}$ and $\boldsymbol{D}$ can be obtained by several diagonalisation algorithms and their computation does not need to be rigorous. Since, $\boldsymbol{\Gamma}_l \in \mathcal{S}_n^+$, one can also use a singular values decomposition to diagonalise. However the computation of $\boldsymbol{M}^+$, $\left[\widetilde{\boldsymbol{b}}\right]$ and ρ must be rigorous.

The parameters $\alpha_1 > 0$ and $\alpha_2 > 0$ are threshold to check if $\det\left(\boldsymbol{\Gamma}_l\right) \neq 0$ and $d_i > 0$. This threshold must be used because of the approximation error in the computation of $\det\left(\boldsymbol{\Gamma}_l\right)$ and d_i, even when they are equal to zero.

A limit of this algorithm is that $[\boldsymbol{s}_i]$ is never fully flat even if the output is flat, because of the pessimism in the computation of $[\boldsymbol{s}_i]$. Using the very small value of $\text{ub}\left([\boldsymbol{s}_i]\right)$ result in horrible pessimism in practice. So, when one has mathematical evidence that $[\boldsymbol{s}_i] = \{0\}$, then one should consider $s_i = 0$. However, this is not feasible on a high-dimensional system when this case occurs. In the applications of this thesis, $[\boldsymbol{s}_i] = \{0\}$ may be assumed true without a full guarantee.

As said before, the parameter q_i can be tuned to reduce the value of ρ to reduce pessimism. One should consider $q_i = 1$ by default. When the values are properly selected, the pessimism of the method is similar to the original one.

7.5 Application

This section illustrates the general propagation on a 2-dimensional example. In this example, a controller relies on low-frequency measurement, creating some delays in the correction action, which can make the system unstable. Consider the hybrid dynamical system described by

$$\begin{aligned} \dot{x} &= \boldsymbol{f}\left(x, m_k\right), \text{for}\, t \in [t_k, t_{k+1}[\\ m_k &= x\left(t_k\right), \end{aligned} \tag{7.48}$$

where $x \in \mathbb{R}$ is a continuous-time variable and $m_k \in \mathbb{R}$ is a discrete-time measurement of x at the time $t_k = T \cdot k$ with $T > 0$. The dynamics of the system are described by the differentiable function

$$\begin{aligned} \boldsymbol{f} : \mathbb{R}^2 &\to \mathbb{R} \\ (x, m) &\mapsto \frac{1}{2}\sin\left(x\right) - m, \end{aligned} \tag{7.49}$$

Since, $\boldsymbol{f}(0,0) = 0$, the origin $(x, m) = (0, 0)$ is an equilibrium point of the hybrid system. Moreover, as illustrated by Figure 7.7, the stability of the origin depends on the value of T. In the function $\boldsymbol{f}$, the term $\frac{1}{2}\sin\left(z\right)$ makes the system diverge, but when m_k is close to x, the term $-m$ compensate the sinus and make the system converge towards 0. So, if m_k is frequently updated (little T), then $x\left(t\right)$ converges towards 0 and m_k follows it. However, when m_k is updated at a low frequency (large T), there are too many delays in the correction, so there is some overshoot and so $x\left(t\right)$ diverges.

The evolution of the system can then be described by a piecewise continuous state vector $\boldsymbol{z}\left(t\right) = \left[\begin{array}{cc} x\left(t\right) & m_k \end{array}\right]^T \in \mathbb{R}^2$ for all $t \in]t_k, t_{k+1}]$ whose evolution is described

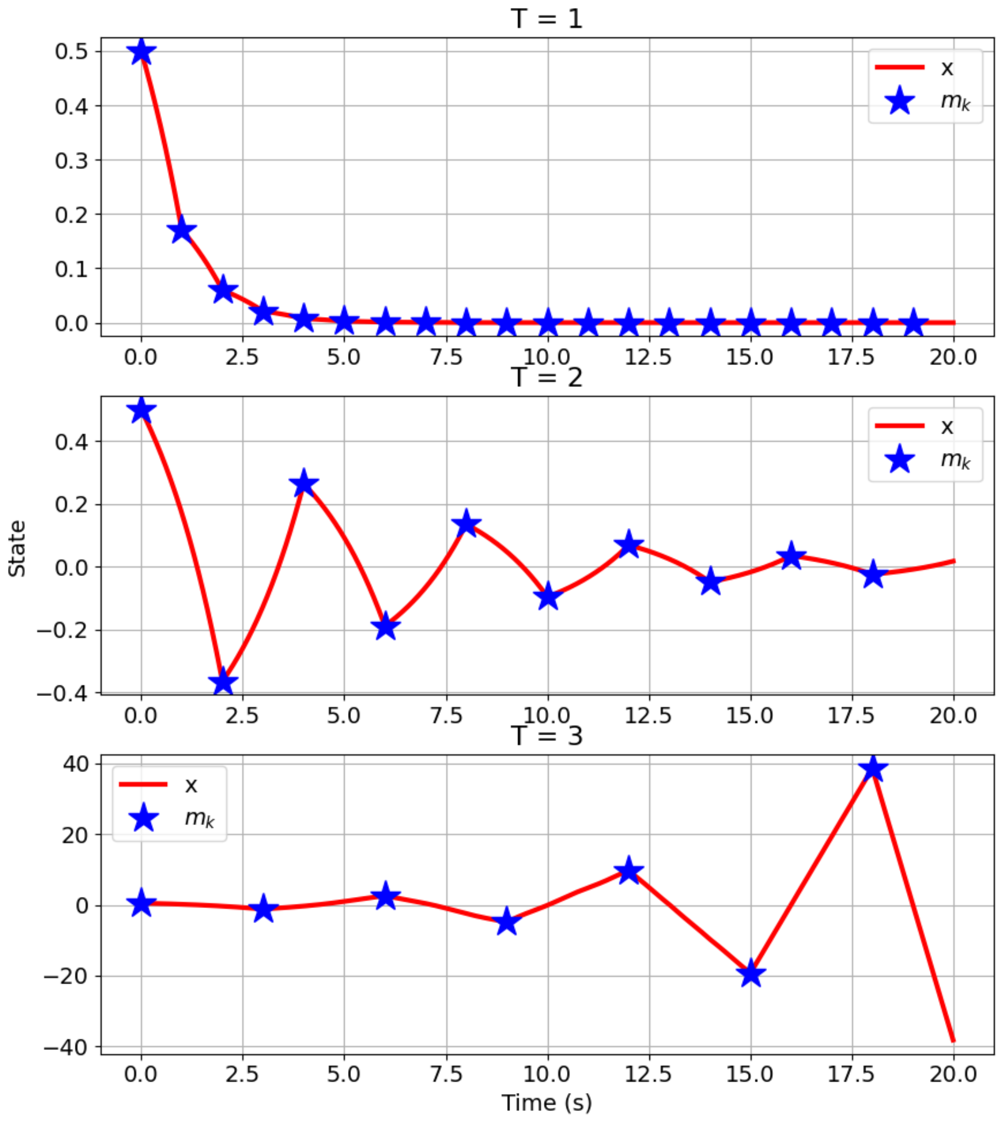


Figure 7.7: Time evolution of the state $(x(t), m_k)$. With a period $T = 1$, the system is stable. With $T = 2$, there is some overshoot but the system is stable. With $T = 3$, the system is unstable.

by

$$\begin{aligned}\dot{\boldsymbol{z}}(t) &= \boldsymbol{f}_z(\boldsymbol{z}(t)),\\ \boldsymbol{z}\left(t_k^+\right) &= \boldsymbol{h}_z(\boldsymbol{z}(t_k)).\end{aligned} \tag{7.50}$$

where

$$\boldsymbol{f}_z(\boldsymbol{z}(t)) = \begin{bmatrix} \boldsymbol{f}(x(t), m_k) \\ 0 \end{bmatrix}, \text{for } t \in]t_k, t_{k+1}] \tag{7.51}$$

and where

$$\boldsymbol{h}_z(\boldsymbol{z}) = \boldsymbol{M} \cdot \boldsymbol{z}, \tag{7.52}$$

with

$$\boldsymbol{M} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}. \tag{7.53}$$

As detailed in the following Chapter 8, to study the stability of this system, it will be discretised at the times t_k, giving the discrete system

$$\boldsymbol{z}(t_{k+1}) = \boldsymbol{\phi}_{z,T} \circ \boldsymbol{h}_z(\boldsymbol{z}(t_k)), \tag{7.54}$$

where $\boldsymbol{\phi}_{z,T}$ is the flow function of $\boldsymbol{f}_z$ for a time T. Let us write $\boldsymbol{h}_{\text{loop}} = \boldsymbol{\phi}_{z,T} \circ \boldsymbol{h}_z$. From (7.49), the origin is an equilibrium point of the hybrid system, i.e. $\boldsymbol{h}_{\text{loop}}(\mathbf{0}) = \mathbf{0}$. Then, consider the ellipsoid

$$\begin{aligned}\mathscr{E} &= \mathcal{E}(\boldsymbol{\Gamma}),\\ \boldsymbol{\Gamma} &= \begin{bmatrix} 0.1 & 0.5 \\ 0.5 & 0.01 \end{bmatrix}.\end{aligned} \tag{7.55}$$

The ellipsoidal enclosure $\mathcal{P}_{h_{\text{loop}}}(\mathscr{E})$ of $\boldsymbol{h}_{\text{loop}}(\mathscr{E})$ is computed with the guaranteed ellipsoidal propagation method. The Jacobian matrix of $\boldsymbol{h}_{\text{loop}}$ is computed from the Jacobian matrix of $\boldsymbol{\phi}_{z,T}$ and $\boldsymbol{h}_z$ such that

$$\frac{\partial \boldsymbol{h}_{\text{loop}}}{\partial \boldsymbol{z}}(\mathscr{E}) = \frac{\partial \boldsymbol{\phi}_{z,T}}{\partial \boldsymbol{z}}(\boldsymbol{h}(\mathscr{E})) \cdot \frac{\partial \boldsymbol{h}_z}{\partial \boldsymbol{z}}(\mathscr{E}). \tag{7.56}$$

The Jacobian matrix of $\boldsymbol{\phi}_{z,T}$ is computed by a guaranteed integration of the variational equation. The Jacobian of $\boldsymbol{h}_z$ is $\boldsymbol{M}$. Since $\boldsymbol{h}_z$ is a singular mapping, $\mathcal{P}_{h_{\text{loop}}}(\mathscr{E})$ is computed will Algorithm 4.

The result of this computation is illustrated by Figure 7.8. The mapping $\boldsymbol{h}_z$ flatten $\mathscr{E}$. But with the effect of, $\boldsymbol{\phi}_{z,T}$, $\boldsymbol{h}_{\text{loop}}(\mathscr{E})$ is likely not flat.

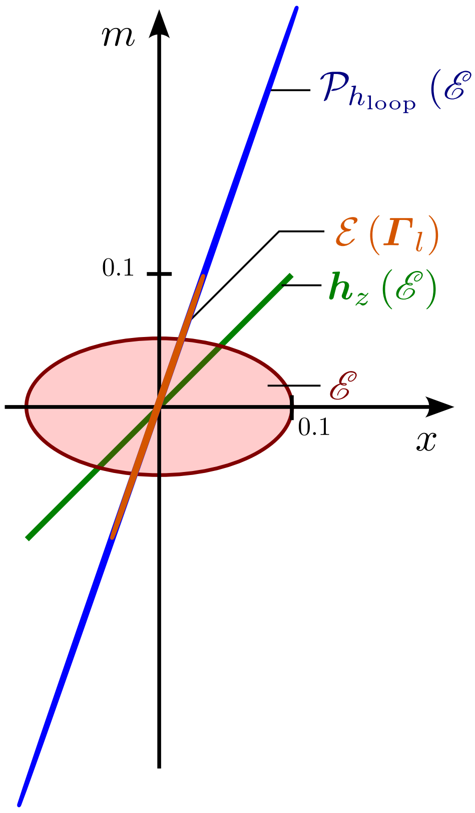


Figure 7.8: Result of the computation of $\mathcal{P}_{h_{\mathrm{loop}}}(\mathscr{E})$

7.6 Conclusion

This chapter presents a generalisation of the guaranteed ellipsoidal to include singular mappings and degenerate ellipsoids. In the singular case, new nonzero eigenvalues, computed with interval analysis, are added to the propagated ellipsoid. As the pessimism of the method is sensitive to the choice of the new eigenvalues, some additional parameter tuning is required. Moreover, when the image of the nonlinear mapping is flat, assumptions on the flatness of the mapping are required for the method to rigorously compute a flat outer ellipsoidal enclosure. The results of this chapter have been presented at the 7th IFAC Conference on Analysis and Control of Nonlinear Dynamics and Chaos [61]. This paper received the Young Author Award. In next following chapters, this generalised method will be used to study the stability of synchronous hybrid nonlinear systems.

Chapter 8

Stability of synchronous Hybrid Systems

8.1 Introduction

After studying the stability of discrete-time systems in Chapter 5, and the stability of continuous-time systems in Chapter 6, this chapter studies their combination: synchronous hybrid systems, presented in Section 3.2.3. Based on the methods presented in Chapter 5 and Chapter 6, this chapter proposes a new method to prove that the system is exponentially stable in an ellipsoid. This chapter also uses the results on the singular mappings and the degenerate ellipsoids, presented in Chapter 7.

As in the previous chapters, this chapter presents a method to compute an ellipsoid in which the system is exponentially stable, in the sense that every trajectory of the system that starts in the ellipsoids converges exponentially towards the equilibrium point. However, this chapter does not provide a method to prove that the ellipsoid is also positive invariant with respect to the system. The topic of positive invariance will be discussed at the conclusion of the chapter.

The stability analysis method of this chapter is similar to the one in Chapter 6 in the sense that the system is discretised. For synchronous hybrid systems, the discretisation is made with cycle mapping, which is the composition of the systems' periodic continuous-time and discrete-time mappings. The stability of the synchronous system is deduced from the stability of the discretised systems. A similar form of deduction has been proposed in [79, 36], using a Lyapunov function. However, as the method of this chapter relies on the guaranteed propagation of ellipsoids, it can be applied to high-dimensional systems.

The chapter is organised as follows. Section 8.2 presents the specificity of degenerate ellipsoids. Section 8.3 present how the stability of the synchronous hybrid systems can be deduced from the stability of the discretised systems. Section 8.4 discusses the algorithmic implementation of the stability analysis. Finally, Section 8.5 applies the method to a hybrid version of the application examples in Chapters 5 and 6.

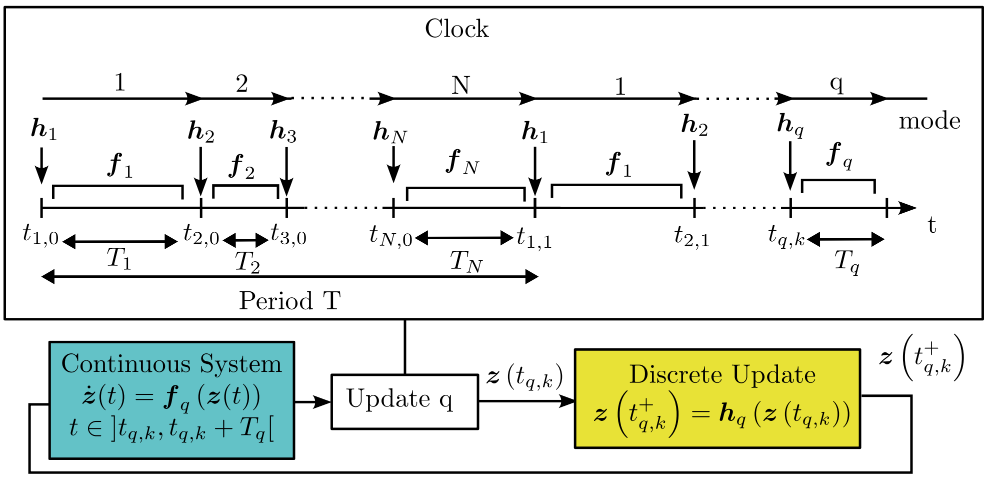


Figure 8.1: The synchronous hybrid system is a continuous-time system with discrete-time updates. Discrete updates are time-dependant.

8.2 Problem definition

Consider a set of modes

$$\mathcal{Q} = \{1, 2, \ldots, N\}, \tag{8.1}$$

a set of increasing times $\{\tau_1, \tau_2, \ldots, \tau_N\} \in \mathbb{R}$ with $\tau_1 = 0$ and a period $T > \tau_N$. Consider the two time representations

$$t_{q,k}, = t_q + k \cdot T, \ \forall q \in \mathcal{Q}, \ \forall k \in \mathbb{N}, \tag{8.2}$$

$$t_i = \tau_{i - \left\lfloor \frac{i-1}{N} \right\rfloor \cdot N} + \left\lfloor \frac{i-1}{N} \right\rfloor \cdot T, \ \forall i \in \mathbb{N}, \tag{8.3}$$

and the time delays

$$T_q = (t_{q+1} - t_q) > 0, \ \forall q \in \mathcal{Q} \tag{8.4}$$

In addition, consider a synchronous hybrid system

$$\begin{cases} \dot{\boldsymbol{z}}(t) & = \boldsymbol{f}_q(\boldsymbol{z}(t)), \text{ if } t \in]t_{q,k}, t_{q,k} + T_q[, \\ \boldsymbol{z}(t_{q,k}^+) & = \boldsymbol{h}_q(\boldsymbol{z}(t_{q,k})), \end{cases} \tag{8.5}$$

illustrated by Figure 8.1, as presented in Section 3.2.3 of Chapter 3, where $t_{q,k}^+$ represent the time instant just after the discrete update, and where, for all $q \in \mathcal{Q}$, the mappings $\boldsymbol{f}_q : \mathbb{R}^p \to \mathbb{R}^p$ is $K_{q,f}$-Lipschitz and the mapping $\boldsymbol{h}_q : \mathbb{R}^p \to \mathbb{R}^p$ is $K_{q,h}$-Lipschitz. So according to Theorem 3.2, these mappings are also $\mathcal{C}^1$. The solution of the system $\boldsymbol{z}(t)$ is piecewise continuous. Assume that $\boldsymbol{z}(t) = \boldsymbol{0}$ is a constant solution of the system, such that for all $q \in \mathcal{Q}$

$$\boldsymbol{0} = \boldsymbol{f}_q(\boldsymbol{0}), \tag{8.6}$$

$$\boldsymbol{0} = \boldsymbol{h}_q(\boldsymbol{0}). \tag{8.7}$$

To evaluate the convergence of $\boldsymbol{z}(t)$, the problem of this chapter consists in computing an ellipsoid $\mathscr{E}$ of initial conditions with which $\boldsymbol{z}(t)$ converge exponentially towards the equilibrium point $\mathbf{0}$, i.e. there exist $\alpha > 0$ and $\gamma \in]0,1[$ that verify

$$\boldsymbol{z}(0) \in \mathscr{E} \Rightarrow \|\boldsymbol{z}(t)\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha \|\boldsymbol{z}(0)\|_{\boldsymbol{\Gamma}^{-2}} e^{t \cdot \ln \gamma}, \forall k \in . \tag{8.8}$$

with $\boldsymbol{\Gamma} = \mathcal{E}^{-1}(\mathscr{E})$. If this ellipsoid exists, then the system is exponentially stable in $\mathscr{E}$. Note that this chapter does not deal with the positive invariance topic which will be discussed at the conclusion of the Chapter. The problem is decomposed into two steps:

- Discretising the system with a cycle mapping to prove its stability with the method of Chapter 5 for the resulting system, as presented in Section 8.3.1.
- Deducing that the system is exponentially stable in the ellipsoid $\mathscr{E}$, as presented in Section 8.3.2

8.3 Proof of the exponential stability

8.3.1 Proving the stability of the discretised system.

The synchronous hybrid system from Section 8.2 can be discretised with respect to the event time $t_{q,k}$, using the cycle mapping presented in Section 3.2.3. To recall, each mode $q \in \mathcal{Q}$ last for a duration $T_q = (t_{i+1,0} - t_{i,0}) > 0$, such that $\sum_{q \in \mathcal{Q}} T_q = T$. So from the beginning to the end of the mode, the state $\boldsymbol{z}$ is affected by the flow $\boldsymbol{\phi}_{q,T_q}$ of $\boldsymbol{f}_q$ for a duration T_q such that

$$\boldsymbol{z}(t_{q+1,k}) = \boldsymbol{\phi}_{q,T_q}\left(\boldsymbol{z}\left(t_{q,k}^+\right)\right). \tag{8.9}$$

As a result, the cycle of the synchronous hybrid system is described by the mapping

$$\boldsymbol{h}_{\text{cycle}} = \boldsymbol{\phi}_{N,T_N} \circ \boldsymbol{h}_N \circ \ldots \circ \boldsymbol{\phi}_{2,T_2} \circ \boldsymbol{h}_2 \circ \boldsymbol{\phi}_{1,T_2} \circ \boldsymbol{h}_1. \tag{8.10}$$

that verifies the following recursive formula

$$\begin{aligned} \boldsymbol{z}_{k+1} &= \boldsymbol{h}_{\text{cycle}}(\boldsymbol{z}_k), \\ \boldsymbol{z}_k &= \boldsymbol{z}(t_{1,k}), \end{aligned} \tag{8.11}$$

illustrated by Figure 8.2. Of course, it is possible to define other $\boldsymbol{h}_{\text{cycle}}$ mappings depending on when the cycle starts. The stability of the discretised system can be studied with the method from Chapter 5, as detailed in Section 8.4.

8.3.2 Extension of stability to the synchronous hybrid system

After proving the exponential stability of the discrete system (8.11), the stability of the synchronous hybrid systems is deduced with Theorem 8.1, illustrated by Figure 8.3.

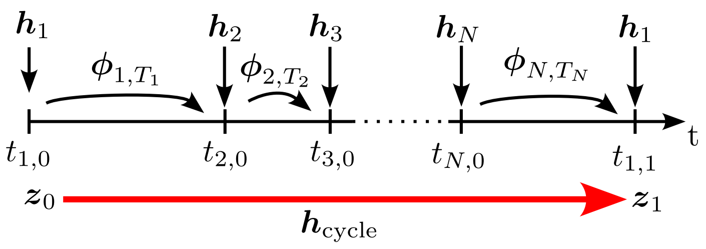
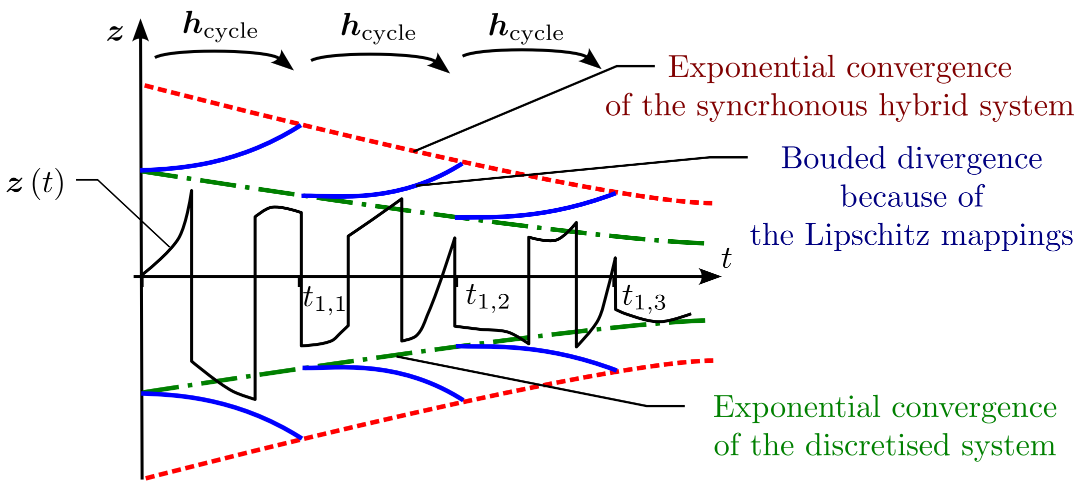


Figure 8.2: Discretisation of the synchronous hybrid system



z
h_{cycle}
h_{cycle}
h_{cycle}
Exponential convergence of the syncrhonous hybrid system
$z(t)$
Bouded divergence because of the Lipschitz mappings
t
$t_{1,1}$
$t_{1,2}$
$t_{1,3}$
Exponential convergence of the discretised system

Figure 8.3: The exponential convergence of the synchronous hybrid system is deduced from the exponential convergence of the discretised system because the mappings composing $\boldsymbol{h}_{\text{cycle}}$ are Lipschitz.

Theorem 8.1. *Consider a nonlinear synchronous hybrid system*

$$\begin{cases} \dot{\boldsymbol{z}}(t) & = \boldsymbol{f}_q(\boldsymbol{z}(t)), \text{ if } t \in]t_{q,k}, t_{q,k} + T_q[, \\ \boldsymbol{z}\left(t_{q,k}^+\right) & = \boldsymbol{h}_q(\boldsymbol{z}(t_{q,k})), \end{cases} \tag{8.12}$$

with the piecewise continuous state $\boldsymbol{z}(t) \in \mathbb{R}^n$, the $K_{q,f}$-Lipschitz mappings $\boldsymbol{f}_q$, the $K_{q,h}$-Lipschitz mappings $\boldsymbol{h}_q$ and time representation $t_{q,k}$ presented in Section 8.2. Consider the cycle mapping of the synchronous hybrid system

$$\boldsymbol{h}_{cycle} = \boldsymbol{\phi}_{N,T_N} \circ \boldsymbol{h}_N \circ \ldots \circ \boldsymbol{\phi}_{2,T_2} \circ \boldsymbol{h}_2 \circ \boldsymbol{\phi}_{1,T_2} \circ \boldsymbol{h}_1, \tag{8.13}$$

and the discretised system

$$\begin{aligned} \boldsymbol{z}_{k+1} &= \boldsymbol{h}_{cycle}(\boldsymbol{z}_k), \\ \boldsymbol{z}_k &= \boldsymbol{z}(t_{1,k}), \end{aligned} \tag{8.14}$$

If the discretised system (8.14) is locally exponentially stable at the origin, then the synchronous hybrid system (8.12) is also locally exponentially stable at the origin.

Proof. Let $\boldsymbol{z}_0 \in \mathbb{R}^p$. From (5.4), since the discretised system is locally exponentially stable, then there exist an ellipsoid $\mathscr{E}$, $\alpha > 0$ and $\gamma \in]0, 1[$ that verify

$$\boldsymbol{z}_0 \in \mathscr{E} \Rightarrow \|\boldsymbol{z}_k\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha \|\boldsymbol{z}_0\|_{\boldsymbol{\Gamma}^{-2}} e^{k \cdot \ln \gamma}, \forall k \in \mathbb{N}, \tag{8.15}$$

$$\Leftrightarrow \|\boldsymbol{z}(t_{1,k})\|_{\boldsymbol{\Gamma}^{-2}} \leq \alpha \|\boldsymbol{z}(0)\|_{\boldsymbol{\Gamma}^{-2}} e^{k \cdot \ln \gamma}, \forall k \in \mathbb{N}, \tag{8.16}$$

with $\boldsymbol{\Gamma} = \mathcal{E}^{-1}(\mathscr{E})$. In addition, for all $q \in \mathcal{Q}$, $\boldsymbol{h}_q$ is $K_{q,h}$-Lipschitz, so from Definition 3.9

$$\begin{aligned} \left\|\boldsymbol{z}\left(t_{q,k}^+\right)\right\|_2 &= \|\boldsymbol{h}(\boldsymbol{z}(t_{q,k}))\|_2 \\ &\leq K_{q,h} \cdot \|\boldsymbol{z}(t_{q,k})\|_2 . \end{aligned} \tag{8.17}$$

Moreover, for all $q \in \mathcal{Q}$, $\boldsymbol{f}_q$ is $K_{q,f}$-Lipschitz, so from Theorem 3.3, one gets

$$\|\boldsymbol{z}(t)\|_2 \leq \left\|\boldsymbol{z}\left(t_{q,k}^+\right)\right\|_2 \cdot e^{K_{q,f}\left(t - t_{q,k}\right)}, \ \forall t \in]t_{q,k}, t_{q+1,k}[. \tag{8.18}$$

As a result, from (8.15), (8.17) and (8.18), one obtains

$$\|\boldsymbol{z}(t)\|_2 \leq \prod_{i=1}^{q} K_{i,h} \cdot \alpha \|\boldsymbol{z}(0)\|_{\boldsymbol{\Gamma}^{-2}} e^{K\left(t - t_{q,k}\right) + k \cdot \ln \gamma} \cdot \forall t \in]t_{q,k}, t_{q+1,k}[\tag{8.19}$$

Then, with $\Delta t = t - t_{q,k} \in [0, T[$ and knowing that $t_{q,k} = \tau_q + k \cdot T$ with $\tau_q \in [0, T[$, one has

$$\begin{aligned} K(t - t_{q,k}) + k \cdot \ln \gamma &= K_{q,f}(t - t_{q,k}) + (t_{q,k} - \tau_q) \cdot \frac{\ln \gamma}{T}, \\ &= K_{q,f} \cdot \Delta t + (t - \Delta t - \tau_q) \cdot \frac{\ln \gamma}{T}, \\ &= \left(K_{q,f} - \frac{\ln \gamma}{T}\right) \Delta t - \tau_q \frac{\ln \gamma}{T} + t \frac{\ln \gamma}{T}. \end{aligned} \tag{8.20}$$

Moreover, since $\gamma \in]0, 1[$ and $(\tau_q, \Delta t) \in [0, T[$, one deduces

$$\begin{aligned} K(t - t_{q,k}) + k \cdot \ln \gamma &< \left(K_{q,f} - \frac{\ln \gamma}{T}\right) T - T \frac{\ln \gamma}{T} + t \frac{\ln \gamma}{T} \\ &= K_{q,f} T - 2 \ln \gamma + t \frac{\ln \gamma}{T}. \end{aligned} \tag{8.21}$$

Thus, one deduces

$$\|\boldsymbol{z}(t)\|_2 \leq \alpha' \|\boldsymbol{z}_0\|_2 e^{t \cdot \ln \gamma'},$$

with

$$\alpha' = \prod_{i=1}^{q} K_{i,h} \cdot \alpha e^{K_{q,f} T - 2 \ln \gamma} > 0 \tag{8.22}$$

and

$$\gamma' = \gamma^{\frac{1}{T}}$$

Therefore, according to (8.8), the synchronous hybrid system is exponentially stable in $\mathscr{E}$. □

8.4 Implementation

The implementation of the stability analysis method presented in Section 8.3 consists in using Algorithm 2 from Chapter 5 on the discretised system

$$\boldsymbol{z}_{k+1} = \boldsymbol{h}_{\text{cycle}}\left(\boldsymbol{z}_k\right). \tag{8.23}$$

To use this algorithm:

- one must compute an approximation of the Jacobian matrix of $\boldsymbol{h}_{\text{cycle}}$ at the origin

$$\boldsymbol{J}_{\text{cycle}} \simeq \frac{\partial \boldsymbol{h}_{\text{cycle}}}{\partial \boldsymbol{z}}\left(0\right). \tag{8.24}$$

- for a given ellipsoid $\mathscr{E}$, one must compute an interval matrix $[\boldsymbol{J}_{\text{cycle}}]$ such that

$$\frac{\partial \boldsymbol{h}_{\text{cycle}}}{\partial \boldsymbol{z}}\left(\mathscr{E}\right) \subseteq \left[\boldsymbol{J}_{\text{cycle}}\right]. \tag{8.25}$$

- When $\boldsymbol{J}_{\text{cycle}}$ is singular, some tuning of the axis-aligned Lyapunov equation may be required.

Again, all these computations can be done numerically with polynomial complexity. Therefore the numerical method is computationally tractable and can be used on high dimensional systems.

Computation of $\boldsymbol{J}_{\text{cycle}}$ The approximation $\boldsymbol{J}_{\text{cycle}}$ can be computed from the Jacobians of the mappings $\boldsymbol{h}_q$ and $\boldsymbol{\phi}_{q,T_q}$ from (8.10). The Jacobians of the $\boldsymbol{h}_q$ at the origin,

$$\boldsymbol{J}_{h_q} = \frac{\partial \boldsymbol{h}_q}{\partial \boldsymbol{z}}\left(\boldsymbol{0}\right), \tag{8.26}$$

can be deduced from the analytical expression of $\boldsymbol{h}_q$. An approximation of the Jacobians of $\boldsymbol{\phi}_{q,T_q}$ at the origin,

$$\boldsymbol{J}_{\phi_{q,T_q}} \simeq \frac{\partial \boldsymbol{\phi}_{q,T_q}}{\partial \boldsymbol{z}}\left(\boldsymbol{0}\right), \tag{8.27}$$

can be evaluated with the integration of the variational equation, as presented in Section 4.5 from Chapter 3. Then, one can compute

$$\boldsymbol{J}_{\text{cycle}} = \boldsymbol{J}_{\phi_{N,T_N}} \cdot \boldsymbol{J}_{h_N} \cdot \ldots \cdot \boldsymbol{J}_{\phi_{2,T_2}} \cdot \boldsymbol{J}_{h_2} \cdot \boldsymbol{J}_{\phi_{1,T_1}} \cdot \boldsymbol{J}_{h_1} \tag{8.28}$$

Algorithm 5 Computation of $[\boldsymbol{J}_{\text{cycle}}]$

```
Inputs $\mathscr{E}$
Outputs $[\boldsymbol{J}_{\text{cycle}}]$
 1: $\mathscr{E}_i = \mathscr{E}$
 2: $[\boldsymbol{J}_{\text{cycle}}] = \boldsymbol{I}_p$
 3: for $q$ from 1 to $N$ do
 4:     $\left[\boldsymbol{J}_{h_q}\right] = \left[\frac{\partial \boldsymbol{h}_q}{\partial \boldsymbol{z}}\right]([\mathscr{E}_i])$
 5:     $\mathscr{E}_i = \mathcal{P}_{h_q}(\mathscr{E}_i)$
 6:
 7:     $\left[\boldsymbol{J}_{\phi_{q,T_q}}\right] = $ integration_variational_equation $\left(\mathscr{E}_i, \boldsymbol{f}_q, T_q\right)$
 8:     // see Section 4.5.2.
 9:
10:     $\mathscr{E}_i = \mathcal{P}_{\phi_{q,T_q}}(\mathscr{E}_i)$
11:     $[\boldsymbol{J}_{\text{cycle}}] = \left[\boldsymbol{J}_{\phi_{q,T_q}}\right] \cdot \left[\boldsymbol{J}_{h_q}\right] \cdot [\boldsymbol{J}_{\text{cycle}}]$
12: end for
```

Computation of $[\boldsymbol{J}_{\text{cycle}}]$ The computation of $[\boldsymbol{J}_{\text{cycle}}]$ is more challenging as it requires a recursive method presented by Algorithm 5.

In this algorithm, an interval matrices of the Jacobian, $\left[\boldsymbol{J}_{h_q}\right]$ and $\left[\boldsymbol{J}_{\phi_{q,T_q}}\right]$ are computed for the intermediate mappings, $\boldsymbol{h}_q$ and $\boldsymbol{\phi}_{q,T_q}$ of (8.10) so that

$$[\boldsymbol{J}_{\text{cycle}}] = \left[\boldsymbol{J}_{\phi_{N,T_N}}\right] \cdot \left[\boldsymbol{J}_{h_N}\right] \cdot \ldots \cdot \left[\boldsymbol{J}_{\phi_{2,T_2}}\right] \cdot \left[\boldsymbol{J}_{h_2}\right] \cdot \left[\boldsymbol{J}_{\phi_{1,T_1}}\right] \cdot \left[\boldsymbol{J}_{h_1}\right]. \tag{8.29}$$

However, these interval matrices must be computed on intermediate sets, such that for each $q \in \mathcal{Q}$, one has

$$\begin{aligned} \frac{\partial \boldsymbol{h}_q}{\partial \boldsymbol{z}}(\mathcal{S}_q) &\subseteq \left[\boldsymbol{J}_{h_q}\right], \\ \frac{\partial \boldsymbol{\phi}_{q,T_q}}{\partial \boldsymbol{z}}\left(\mathcal{S}_q^{'}\right) &\subseteq \left[\boldsymbol{J}_{\boldsymbol{\phi}_{q,T_q}}\right], \end{aligned} \tag{8.30}$$

with the intermediate sets

$$\mathcal{S}_q = \boldsymbol{\phi}_{q-1,T_{q-1}} \circ \boldsymbol{h}_{q-1} \circ \ldots \circ \boldsymbol{\phi}_{1,T_1} \circ \boldsymbol{h}_1(\mathscr{E}), \tag{8.31}$$

$$\mathcal{S}_q^{'} = \boldsymbol{h}_q(\mathcal{S}_q). \tag{8.32}$$

which contains the state $\boldsymbol{z}(t)$ at the intermediate times. In the algorithm, the sets $\mathcal{S}_q$ and $\mathcal{S}_q^{'}$ are enclosed by the intermediate ellipsoid $\mathscr{E}_i$, on which the Jacobians are evaluated. The interval matrix $\left[\boldsymbol{J}_{h_q}\right]$ is computed on $\mathscr{E}_i$ with using the analytical expression of $\frac{\partial \boldsymbol{h}_q}{\partial \boldsymbol{z}}$. The interval matrix $\left[\boldsymbol{J}_{\boldsymbol{\phi}_{q,T_q}}\right]$ is computed on $\mathscr{E}_i$ by a guaranteed integration of the variational equation, as presented in Section 4.5.2.

The intermediate ellipsoid is computed with the guaranteed propagation of ellipsoid, with the operators $\mathcal{P}_{h_q}$ and $\mathcal{P}_{\phi_{q,T_q}}$. This is possible because the Jacobian of the intermediate mappings are computed before each propagation. Note that, while $\mathscr{E}_i$ is evaluated by a recursive propagation in Algorithm 5, there exist an alternative method: $\mathscr{E}_i$ can be computed by propagating $\mathscr{E}$ with several intermediate mappings in one step. For example, with (8.31), one can compute

$$\mathscr{E}_i = \mathcal{P}_{\phi_{q-1,T_{q-1}} \circ h_{q-1} \circ \ldots \circ \phi_{1,T_1} \circ h_1} (\mathscr{E}) \tag{8.33}$$

so that $\mathcal{S}_q \subseteq \mathscr{E}_i$, using the intermediate interval matrix

$$[\boldsymbol{J}_i] = \left[\boldsymbol{J}_{\phi_{q-1,T_{q-1}}}\right] \cdot \left[\boldsymbol{J}_{h_{q-1}}\right] \cdot \ldots \cdot \left[\boldsymbol{J}_{\phi_{1,T_1}}\right] \cdot \left[\boldsymbol{J}_{h_1}\right].$$

In this thesis, no detailed studies have been made to compare the performance of these two computations of $\mathscr{E}_i$ in term of computational complexity and pessimism.

As a result, with Algorithm 5 and the Algorithm 4 from Chapter 7, the computation of $\mathcal{P}_{h_{\text{cycle}}}$ requires $2N$ intermediates propagation of ellipsoids, used to evaluate the Jacobian of h_{cycle}.

Axis aligned Lyapunov equation with singularity As presented in Section 5.4, one operation of the Algorithm 2 consist in computing $\boldsymbol{P} \in \mathcal{S}_n^+$ solution of

$$\boldsymbol{J}_{\text{cycle}}^T \boldsymbol{P} \boldsymbol{J}_{\text{cycle}} - \boldsymbol{P} = -\boldsymbol{J}_{\text{cycle}}^T \boldsymbol{J}_{\text{cycle}}. \tag{8.34}$$

Then, in Algorithm 2, the initial ellipsoid is chosen as

$$\mathscr{E} = \mathcal{E}\left(10^{-\alpha} \cdot \boldsymbol{P}^{-\frac{1}{2}}\right)$$

with $\alpha > 0$. If $\boldsymbol{J}_{\text{cycle}}$ is non-singular, then $\mathscr{E}$ is non-degenerate, which is convenient to test the inclusion $\mathcal{P}_{h_{\text{cycle}}}(\mathscr{E}) \subset \mathscr{E}$ with a numerical method. However, if $\boldsymbol{J}_{\text{cycle}}$ is singular, then $\mathscr{E}$ is degenerate and the inclusion is not possible.

Therefore, when $\boldsymbol{J}_{\text{cycle}}$ is singular, the right term of 8.34 can be changed to make $\mathscr{E}$ non-degenerate. For example, $\boldsymbol{P} \in \mathcal{S}_n^+$ can be the solution of

$$\boldsymbol{J}_{\text{cycle}}^T \boldsymbol{P} \boldsymbol{J}_{\text{cycle}} - \boldsymbol{P} = -\boldsymbol{J}^T \boldsymbol{J}. \tag{8.35}$$

where $\boldsymbol{J}$ is a non-singular matrix close to $\boldsymbol{J}_{\text{cycle}}$.

8.5 Application on the formation control

In this section, the stability analysis of synchronous hybrid system is illustrated with a hybrid version of the application example of Chapter 5 and Chapter 6. To recall, two ROVs are controlled with a virtual structure approach to reach an equilateral triangular formation. While this system was represented as a discrete-time system in Chapter 5, and a continuous-time system in Chapter 6, it is now modelled as a synchronous hybrid system.

This new model can thus describe the physical dynamical evolution of the robots while considering the discrete-time measurements and the memory of the embedded computers.

In addition, some modifications have been considered in the model in order to better fit with the real model of Chapter 9:

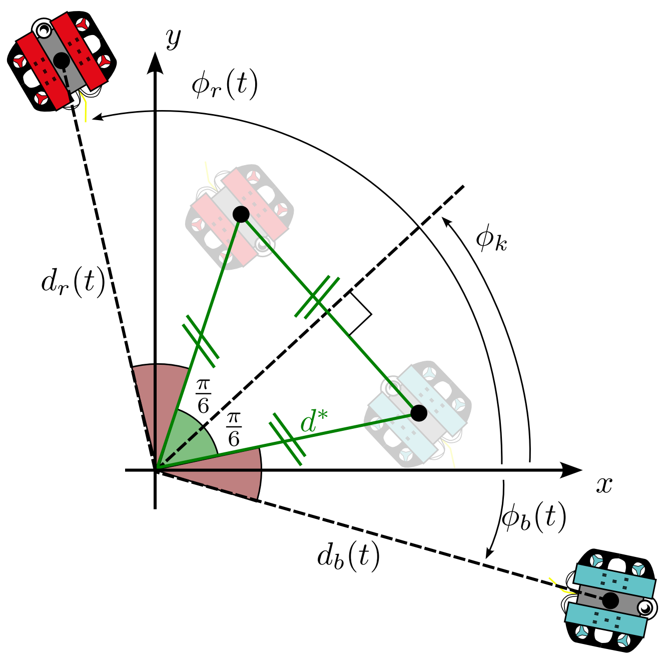


Figure 8.4: Illustration of the two ROVs system with continuous and discrete variables.

- The derivative component of the controller has been removed because the real robots do not measure their speed.
- Damping has been added to the model because damping effects are non-neglectable on the real system. In this section, the damping is only linear but one can also add nonlinear damping.

8.5.1 Mathematical description of the system

As illustrated by Figure 8.4 the ROVs Inky and Blinky are described using polar coordinates. The position and speed of the robots are continuous-time variables. The polar horizontal coordinates of the ROVs are written

$$(d_b, \phi_b) \in \mathbb{R}^2 (\text{for Inky}),$$
$$(d_r, \phi_r) \in \mathbb{R}^2 (\text{for Blinky}).$$

Discrete time measurement The robots have a synchronous clock, but they do not measure their position at the same time, as illustrated by Figure 8.5. Consider two mods

$$q_b = 1 \qquad (\text{After Inky's measurement}), \tag{8.36}$$
$$q_r = 2 \qquad (\text{After Blinky's measurement}), \tag{8.37}$$

and the set of mods $\mathcal{Q} = \{q_b, q_r\}$. Consider the measurement period $T > 0$ and the times $\tau_{q_b} = 0$ and $\tau_{q_r} = \frac{T}{2}$. For all $q \in \mathcal{Q}$, the transition to the mode q happens at

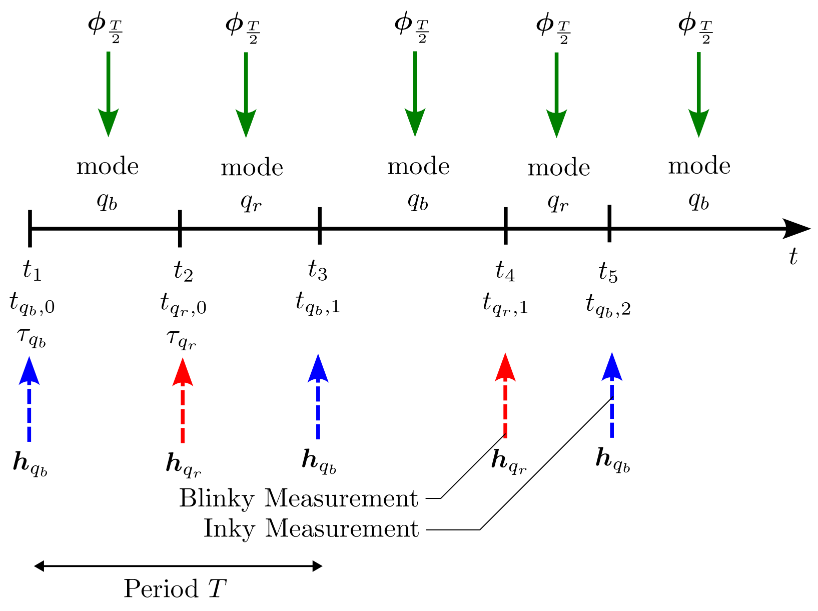


Figure 8.5: Time representation of the system

time

$$t_{q,k} = t_q + k \cdot T, \ \forall q \in \mathcal{Q}, \ \forall k \in \mathbb{Z}. \tag{8.38}$$

The two mods last for a duration $\frac{T}{2}$. The time of the discrete events is also represented with a single indent number such that, for all $j \in \mathbb{N}$,

$$t_j = t_{\left(j - \left\lfloor \frac{j-1}{2} \right\rfloor \cdot 2\right), \left(\left\lfloor \frac{j-1}{2} \right\rfloor\right)} \tag{8.39}$$

$$= (j-1) \cdot \frac{T}{2} \tag{8.40}$$

The measures are memorised in the discrete-time variables $(d_{b,k}, \phi_{b,k}, d_{r,k}, \phi_{r,k})$ such that

$$\begin{aligned} (d_{b,j+1}, \phi_{b,j+1}) &= \begin{cases} (d_{b,j}, \phi_{b,j}) & \text{if } j \text{ is even} \\ (d_b(t_{j+1}), \phi_b(t_{j+1})) & \text{if } j \text{ is odd} \end{cases} \\ (d_{r,i+1}, \phi_{r,i+1}) &= \begin{cases} (d_r(t_{j+1}), \phi_r(t_{j+1})) & \text{if } j \text{ is even} \\ (d_{r,j}, \phi_{r,j}) & \text{if } j \text{ is odd} \end{cases} \end{aligned} \tag{8.41}$$

Motion and control Then, the motion of the robots is described by the following ODE

$$\begin{cases} \ddot{d}_b(t) &= u_{1,j} - c \cdot \dot{d}_b(t), \\ \ddot{d}_r(t) &= u_{2,j} - c \cdot \dot{d}_r(t), \\ \ddot{\phi}_b(t) &= \frac{u_{3,j}}{d_b(t)} - c \cdot \dot{\phi}_b(t), \\ \ddot{\phi}_r(t) &= \frac{u_{4,j}}{d_r(t)} - c \cdot \dot{\phi}_r(t), \end{cases} \forall t \in]t_j, t_{j+1}[\tag{8.42}$$

where $(u_{1,j}, u_{2,j}, u_{3,j}, u_{4,j}) \in \mathbb{R}^4$ are the discrete-time acceleration inputs of the system and where $c > 0$ is a damping coefficient. As in the previous chapters, there is a singularity when $d_b(t) = 0$ or $d_r(t) = 0$ which is avoided in the stability analysis. Then, as illustrated by Figure 8.4, on the time interval $]t_j, t_{j+1}[$, the discrete-time desired positions of the robot are

$$\left(d^*, \phi_j - \frac{\pi}{6}\right)^T \text{(for Inky)}, \tag{8.43}$$

$$\left(d^*, \phi_j + \frac{\pi}{6}\right)^T \text{(for Blinky)}, \tag{8.44}$$

with the desired distance $d^* > 0$ and where the orientation of the triangle ϕ_j is given by

$$\phi_j = \frac{\phi_{b,j} + \phi_{r,j}}{2}. \tag{8.45}$$

The robots track their desired position with the following saturated proportional controller

$$\begin{aligned} u_{1,j} &= s \cdot \arctan\left(k_{p,d} \cdot (d^* - d_{b,j})\right), \\ u_{2,j} &= s \cdot \arctan\left(k_{p,d} \cdot (d^* - d_{r,j})\right), \\ u_{3,j} &= s \cdot \arctan\left(k_{p,\phi} \cdot \left(\phi_j - \frac{\pi}{6} - \phi_{b,j}\right)\right) \\ u_{4,j} &= s \cdot \arctan\left(k_{p,\phi} \cdot \left(\phi_j + \frac{\pi}{6} - \phi_{r,j}\right)\right) \end{aligned} \tag{8.46}$$

with the saturation amplitude $s > 0$ and the controller gains $(k_{p,d}, k_{p,\phi}) > 0$.

Towards the synchronous hybrid model Consider the continuous-time state vector $\boldsymbol{x} = (x_i)_{i \in [\![1:7]\!]} \in \mathbb{R}^7$ with

$$\begin{aligned} x_1 &= d_b - d^*, \\ x_2 &= d_r - d^*, \\ x_3 &= \phi_r - \phi_b - \frac{\pi}{3}, \\ x_4 &= \dot{d}_b, \\ x_5 &= \dot{d}_r, \\ x_6 &= \dot{\phi}_b, \\ x_7 &= \dot{\phi}_r, \end{aligned} \tag{8.47}$$

the discrete-time numerical memory vector $\boldsymbol{m}_j = (m_{i,j})_{i \in [\![1:3]\!]} \in \mathbb{R}^3$ with

$$\begin{aligned} m_{1,j} &= d_{b,j} - d^*, \\ m_{2,j} &= d_{r,j} - d^*, \\ m_{3,j} &= \phi_{r,j} - \phi_{b,j} - \frac{\pi}{3}. \end{aligned} \tag{8.48}$$

and the piecewise continuous error vector $\boldsymbol{v}_j = (v_{i,j})_{i\in[\![1:2]\!]} \in \mathbb{R}^2$ with

$$\begin{aligned} v_1(t) &= \phi_b(t) - \phi_{b,j},\ \forall t \in \left[t_j^+, t_{j+1}\right], \\ v_2(t) &= \phi_r(t) - \phi_{r,j},\ \forall t \in \left[t_j^+, t_{j+1}\right]. \end{aligned} \tag{8.49}$$

The evolution of the continuous state vector is given by the ODE

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}_x(\boldsymbol{x}(t), \boldsymbol{m}_j), \forall t \in]t_j, t_{j+1}[, \\ \mathbf{0} &= \boldsymbol{f}_x(\mathbf{0}, \mathbf{0}), \end{aligned} \tag{8.50}$$

with

$$\boldsymbol{f}_x(\boldsymbol{x}(t), \boldsymbol{m}_j) = \begin{bmatrix} x_4 \\ x_5 \\ (x_7 - x_6) \\ -s \cdot \arctan(k_{p,d} \cdot m_{1,j}) - c \cdot x_4 \\ -s \cdot \arctan(k_{p,d} \cdot m_{2,j}) - c \cdot x_5 \\ \frac{1}{x_1+d^*} s \cdot \arctan\left(k_{p,\phi} \cdot \frac{m_{3,j}}{2}\right) - c \cdot x_6 \\ \frac{-1}{x_2+d^*} s \cdot \arctan\left(k_{p,\phi} \cdot \frac{m_{3,j}}{2}\right) - c \cdot x_7 \end{bmatrix}. \tag{8.51}$$

The computation of $\boldsymbol{f}_x$ is detailed in Appendix 10. The evolution of the memory vector is given by

$$\boldsymbol{m}_{j+1} = \begin{cases} \boldsymbol{h}_{m,q_b}(\boldsymbol{x}(t_{j+1}), \boldsymbol{m}_j, \boldsymbol{v}(t_{j+1})), & \text{if } j \text{ is odd} \\ \boldsymbol{h}_{m,q_r}(\boldsymbol{x}(t_{j+1}), \boldsymbol{m}_j, \boldsymbol{v}(t_{j+1})), & \text{if } j \text{ is even} \end{cases}$$

$$\mathbf{0} = \boldsymbol{h}_{m,q_b}(\mathbf{0}, \mathbf{0}), \tag{8.52}$$

$$\mathbf{0} = \boldsymbol{h}_{m,q_r}(\mathbf{0}, \mathbf{0}), \tag{8.53}$$

with

$$\boldsymbol{h}_{m,q_b}(\boldsymbol{x}(t_{j+1}), \boldsymbol{m}_j, \boldsymbol{v}(t_{j+1})) = \begin{bmatrix} x_1(t_{j+1}) \\ m_{2,j} \\ x_3(t_{j+1}) - v_2(t_{j+1}) \end{bmatrix}, \tag{8.54}$$

and

$$\boldsymbol{h}_{m,q_r}(\boldsymbol{x}(t_{k+1}), \boldsymbol{m}_k, \boldsymbol{v}(t_{j+1})) = \begin{bmatrix} m_{1,j} \\ x_2(t_{j+1}) \\ x_3(t_{j+1}) + v_1(t_{j+1}) \end{bmatrix}.$$

The computation of $\boldsymbol{h}_{m,q_b}$ and $\boldsymbol{h}_{m,q_r}$ are detailed in Appendix 10. Moreover, the evolution of $, \boldsymbol{v}(t)$ is given by

$$\begin{aligned} \dot{\boldsymbol{v}}(t) &= \boldsymbol{f}_v(\boldsymbol{x}(t)), \forall t \in]t_j, t_{j+1}[, \\ \mathbf{0} &= \boldsymbol{f}_v(\mathbf{0}) \\ \boldsymbol{v}\left(t_j^+\right) &= \begin{cases} \boldsymbol{h}_{v,q_b}(\boldsymbol{v}(t_j)), & \text{if } j \text{ is odd,} \\ \boldsymbol{h}_{v,q_r}(\boldsymbol{v}(t_j)), & \text{if } j \text{ is even,} \end{cases} \\ \mathbf{0} &= \boldsymbol{h}_{v,q_b}(\mathbf{0}), \\ \mathbf{0} &= \boldsymbol{h}_{v,q_r}(\mathbf{0}), \end{aligned}$$

with

$$\boldsymbol{f}_v\left(\boldsymbol{x}(t)\right)=\left[\begin{array}{c} x_6(t) \\ x_7(t) \end{array}\right], \tag{8.55}$$

$$\boldsymbol{h}_{v,q_b}\left(\boldsymbol{v}(t_j)\right)=\left[\begin{array}{c} 0 \\ v_2(t_j) \end{array}\right], \tag{8.56}$$

$$\boldsymbol{h}_{v,q_r}\left(\boldsymbol{v}(t_j)\right)=\left[\begin{array}{c} v_1(t_j) \\ 0 \end{array}\right]. \tag{8.57}$$

The computation of $\boldsymbol{f}_v$, $\boldsymbol{h}_{v,q_b}$ and $\boldsymbol{h}_{v,q_r}$ are detailed in Appendix 10.

Synchronous hybrid model To describe the system like (8.5), the variables are regrouped in the state variable

$$\boldsymbol{z}(t)=\left[\begin{array}{c} \boldsymbol{x}(t) \\ \boldsymbol{m}(t) \\ \boldsymbol{v}(t) \end{array}\right]\in\mathbb{R}^{12}, \tag{8.58}$$

with

$$\boldsymbol{m}(t)=\boldsymbol{m}_j,\ \forall t\in\left[t_j^+,t_{j+1}\right]. \tag{8.59}$$

The evolution of this state is deduced from $\boldsymbol{f}_x$, $\boldsymbol{h}_{m,q_b}$, $\boldsymbol{h}_{m,q_r}$, $\boldsymbol{f}_v$, $\boldsymbol{h}_{v,q_b}$ and $\boldsymbol{h}_{v,q_r}$, and is given by

$$\begin{cases} \dot{\boldsymbol{z}}(t) & =\boldsymbol{f}\left(\boldsymbol{z}(t)\right),\ \text{if } t\in\left]t_{q,k},t_{q,k}+\frac{T}{2}\right[, \\ \boldsymbol{z}\left(t_{q,k}^+\right) & =\boldsymbol{h}_q\left(\boldsymbol{z}(t_{q,k})\right), \end{cases} \tag{8.60}$$

where

$$\boldsymbol{f}\left(\boldsymbol{z}(t)\right)=\left[\begin{array}{c} \boldsymbol{f}_x\left(\boldsymbol{x}(t),\boldsymbol{m}_j\right) \\ \mathbf{0} \\ \boldsymbol{f}_v\left(\boldsymbol{x}(t)\right) \end{array}\right], \tag{8.61}$$

$$\boldsymbol{h}_{q_b}\left(\boldsymbol{z}(t_{q_b,k})\right)=\left[\begin{array}{c} \boldsymbol{x}(t_{q_b,k}) \\ \boldsymbol{h}_{m,q_b}\left(\boldsymbol{x}(t_{q_b,k}),\boldsymbol{m}_j,\boldsymbol{v}(t_{q_b,k})\right) \\ \boldsymbol{h}_{v,q_b}\left(\boldsymbol{v}(t_{q_b,k})\right) \end{array}\right], \tag{8.62}$$

$$\boldsymbol{h}_{q_r}\left(\boldsymbol{z}(t_{q_r,k})\right)=\left[\begin{array}{c} \boldsymbol{x}(t_{q_r,k}) \\ \boldsymbol{h}_{m,q_r}\left(\boldsymbol{x}(t_{q_r,k}),\boldsymbol{m}_j,\boldsymbol{v}(t_{q_r,k})\right) \\ \boldsymbol{h}_{v,q_r}\left(\boldsymbol{v}(t_{q_r,k})\right) \end{array}\right]. \tag{8.63}$$

All these mappings are Lipschitz and verify

$$\begin{aligned} \mathbf{0} &= \boldsymbol{f}_q(\mathbf{0}), \\ \mathbf{0} &= \boldsymbol{h}_q(\mathbf{0}). \end{aligned} \tag{8.64}$$

The flow of the mapping $\boldsymbol{f}_q$ for a time $\frac{T}{2}$ is written $\boldsymbol{\phi}_{\frac{T}{2}}$. The cycle mapping given by

$$\boldsymbol{h}_{\text{cycle}}=\boldsymbol{\phi}_{\frac{T}{2}}\circ\boldsymbol{h}_{q_r}\circ\boldsymbol{\phi}_{\frac{T}{2}}\circ\boldsymbol{h}_{q_b}. \tag{8.65}$$

Finally, the parameter values are given by table 8.1. These parameters approximate the real dynamics of the robot of Chapter 9.

Parameter	value	unit
T	9	s
s	2.5	m.s^{-2}
$k_{p,d}$	0.1	m^{-1}
c	1	s^{-1}
$k_{p,\phi}$	0.1	rad^{-1}
d^{*}	3	m

Table 8.1: Parameters of the system

8.5.2 Results

Using the procedure detailed in Section (8.4), an ellipsoid $\mathscr{E} \subseteq \mathbb{R}^{12}$ is computed as well as its propagation $\mathcal{P}_{h_{\text{cycle}}}(\mathscr{E})$. Note that some singular mappings ($\boldsymbol{h}_{v,q_b}$ and $\boldsymbol{h}_{v,q_r}$) are involved in the computation of $\mathcal{P}_{\boldsymbol{h}_{\text{cycle}}}(\mathscr{E})$, so the generalised propagation method of Chapter 7 is used. Then, the inclusion $\mathcal{P}_h(\mathscr{E}) \subset \mathscr{E}$ is verified, as illustrated by Figure (8.6). As a result, $\mathscr{E}$ is positive invariant with respect to the discretised system

$$\boldsymbol{z}_{k+1} = \boldsymbol{h}_{\text{cycle}}(\boldsymbol{z}_k) \tag{8.66}$$

and the synchronous hybrid system (8.60) is exponentially stable in the ellipsoid $\mathscr{E}$.

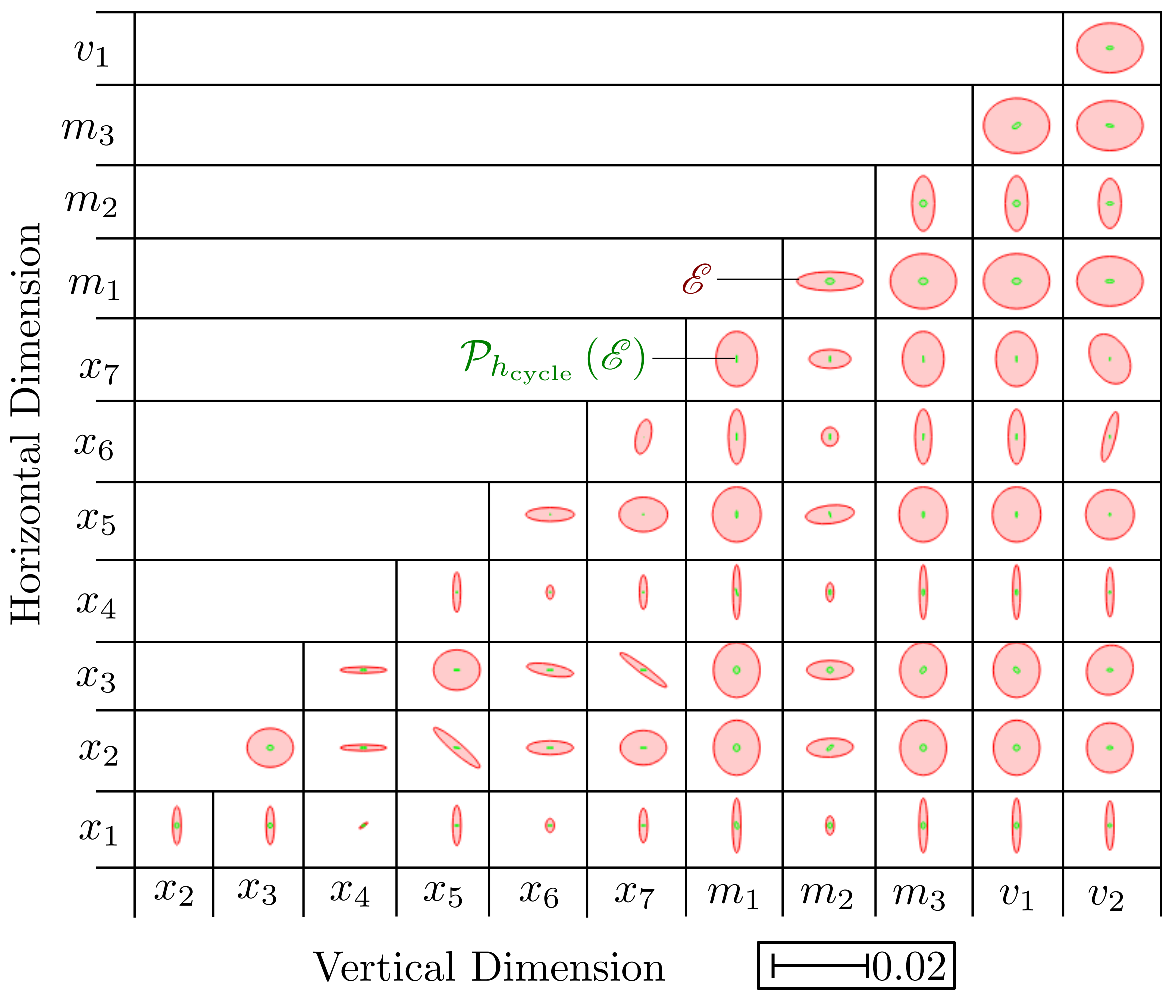


Figure 8.6: Representation of the 12-dimensional ellipsoids $\mathscr{E}$ (red) and $\mathcal{P}_{h_{\text{cycle}}}(\mathscr{E})$ (green) with orthogonal projections on the 2-dimensional planes $(0, x_i, x_j)$ with $i < j$.

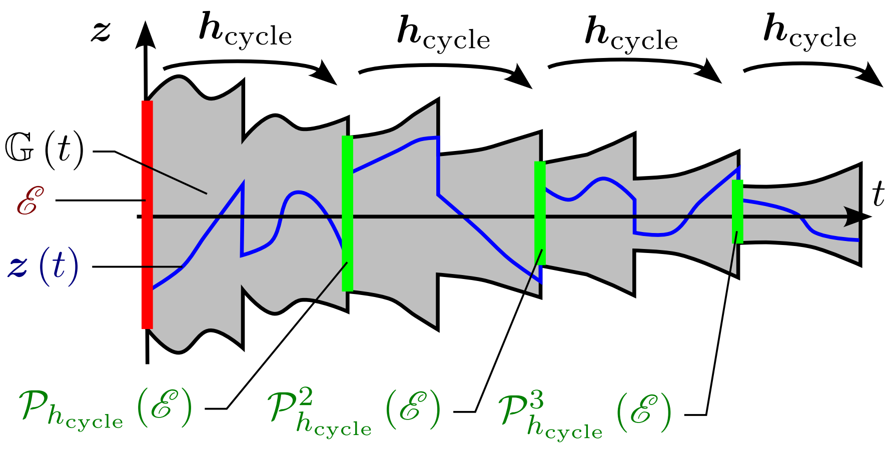
8.6 Conclusion

This chapter presents a numerical guaranteed method to compute ellipsoids in which a high-dimensional nonlinear synchronous hybrid system is stable. This method is based on the results of Chapters 5, 6 and 7. To our knowledge, it is the first time that the stability of such systems has been studied with an interval method.

The stability of the hybrid system is deduced from the stability of a discretised system with a cycle mapping. The method also proves that the ellipsoid is positive invariant with respect to the discretised system.

However, the method does not prove that the ellipsoid is positive invariant with respect to the hybrid system. In other words, the state of the hybrid system is periodically present in the ellipsoid, but the state may escape the ellipsoid during the cycles.

Future work could study the positive invariance for synchronous hybrid systems. Even if the state is not bounded by an ellipsoid at every time, there exist some



$\boldsymbol{z}$
$\boldsymbol{h}_{\text{cycle}}$
$\boldsymbol{h}_{\text{cycle}}$
$\boldsymbol{h}_{\text{cycle}}$
$\boldsymbol{h}_{\text{cycle}}$
$\mathbb{G}(t)$
$\mathscr{E}$
$\boldsymbol{z}(t)$
t
$\mathcal{P}_{h_{\text{cycle}}}(\mathscr{E})$
$\mathcal{P}^2_{h_{\text{cycle}}}(\mathscr{E})$
$\mathcal{P}^3_{h_{\text{cycle}}}(\mathscr{E})$

Figure 8.7: The method of this chapter can prove that $\boldsymbol{z}_{k+1} = \boldsymbol{h}_{\text{cycle}}(\boldsymbol{z}_k)$ is exponentially stable. However, it does not prove that the initial ellipsoid $\mathscr{E}$ is positive invariant. With the additional numerical method, it could be possible to compute a tube $\mathbb{G}(t)$ that enclose all the trajectories $\boldsymbol{z}(t)$ starting in the ellipsoid $\mathscr{E}$.

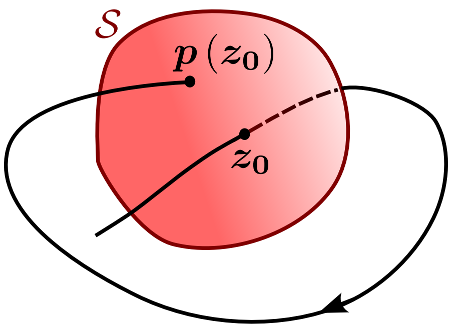


Figure 8.8: In the Poincare Section $\mathcal{S}$, the Poincare mapping $\boldsymbol{p}$ projects the point $\boldsymbol{z}_0$ on $\boldsymbol{p}(\boldsymbol{z}_0)$.

numerical tools that can compute tubes to do so, such as the *codac* library [97], as illustrated by Figure 6.4.

Finally, this method can be extended to the asynchronous hybrid system. These systems are more complex to study, considering that the time of the discrete updates can be state-dependent and change with the initial condition. To deal with the variation of the discrete update time, the hybrid system can be discretised with a Poincare method, presented in [35]. With this complex discretisation, the system is described by mappings, called Poincare mappings, defined on sections of $\mathbb{R}^n$ that are transverse to the flow of the continuous dynamics, as illustrated by Figure 8.8. These sections are often the guards of the hybrid system. There exists some numerical method to compute the Jacobian of a Poincare mapping, as in [42]. Thus It is possible to adapt the stability analysis method to a Poincare discretisation, to study an asynchronous hybrid system.

Chapter 9

Experimental application

In addition to stability analysis, experiments were conducted to illustrate the stability of formation control in a real scenario. This chapter presents a real-world implementation of the application example of Chapter 8. The triangular formation has been experimented with two ROVs from ENSTA Bretagne. The localisation and the control are centralised. Before implementing the real system, the theoretical stability had to be proved in Chapter 8. Details about the experimental set-up are presented in Section 9.1. The results of the experiments are presented in Section 9.2. Discussion about the stability of the experiments is presented in Section 9.3.

9.1 Presentation of the experimental set-up

9.1.1 The ROVs and the USBL

This section presents the robotic materials used during the experiments. Figure 9.1 gives an overview of the experimental set-up. The two ROVs *Inky* and *Blinky* are connected by cable to the same laptop computer that centralises the control and the information. The laptop is also connected to a buoy equipped which can localise the robots with an Ultra Short Baseline (USBL) acoustic sensor.

The ROVs, depicted in Figure 9.2 are *BlueROV2* from the *BlueRobotics* company. These low-cost commercial robots are very common in the academic field and are well-documented online[1]. They have the same thrusters configuration, called *'heavy'*, with 8 thrusters that control their 6 degrees of freedom. They are equipped with:

- a Raspberry Pi 4 Computer.
- a frontal monocular camera[2] used to record videos of the experiment.
- a barometer[3] used to measure the robots' depth.

[1]https://bluerobotics.com/store/bluerov2/
[2]https://bluerobotics.com/store/sensors-sonars-cameras/cameras/cam-usb-low-light-r1/
[3]https://bluerobotics.com/store/sensors-sonars-cameras/sensors/bar30-sensor-r1/

- an IMU[4] to measure their orientation and their heading but it is not accurate enough to provide forward speed. The robots' IMUs are used in the ROVs' attitude control, but not in their position control.
- a *X150 micro-USBL* beacon from the *Blueprint Subsea* company[5]. This model of USBL has a delay of minimum 4 s between two measurements.

A dynamical model of the BlueROV as well as some control advice are presented in [103]. The robots are programmed along with the ROS2 middleware. Moreover, they have translation and rotation commands provided by the MAVROS package. The thrusters produce a maximum translation thrust of about 80 N, which allows the vehicle to reach a maximum horizontal velocity of about $1.5\,\mathrm{m.s}^{-1}$. Forward and lateral inputs must be provided in PWM whose value is between 1000 and 2000, where 1500 is the neutral input, and 1000 and 2000 are the maximum power in one direction or the other. However, the motors have a dead band such that they are inactive if one input is in the interval $[1450, 1550]$.

The USBL buoy, also depicted in Figure 9.2 is a prototype that was designed at the ENSTA Bretagne to localise the two robots. It is equipped with an X150 micro-USBL beacon and a *Razor* IMU[6]. The USBL is configured to send ping to the USBL beacon of the ROVs, to measure their position in its own frame. The IMU allows transforming the robot's position into the North-oriented global frame.

[4]https://bluerobotics.com/store/comm-control-power/control/navigator/

[5]https://www.blueprintsubsea.com/seatrac/seatrac-lightweight

[6]https://learn.sparkfun.com/tutorials/9dof-razor-imu-m0-hookup-guide/all

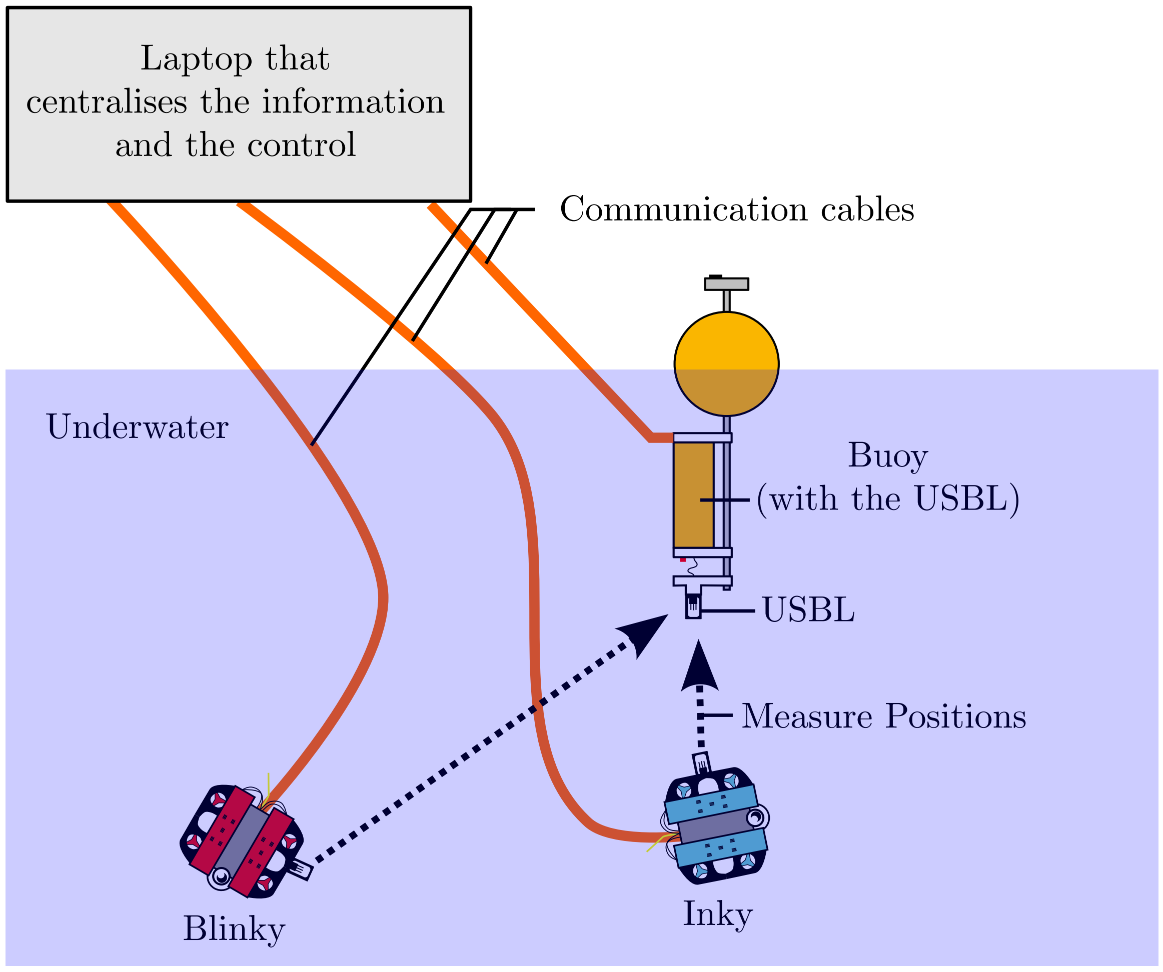


Figure 9.1: Experimental set-up with the two ROVs and the USBL buoy.

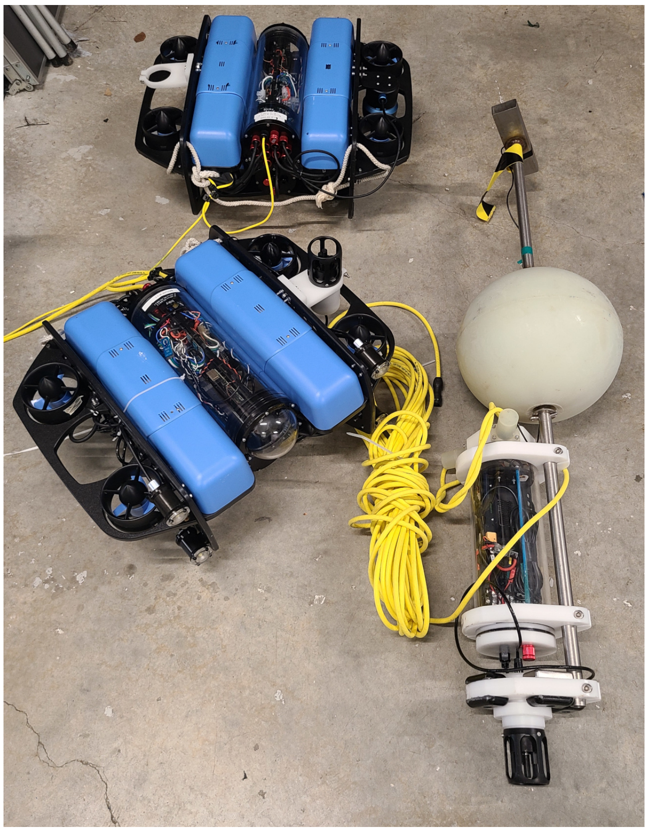


Figure 9.2: The ROVs *Inky* (bottom) and *Blinky* (top) with the USBL buoy

9.1.2 The centralised formation control

The formation control is described in Figure 9.4. Following the application of Chapter 8, the robots are controlled to form an equilateral triangle with the Buoy. The controller is presented in detail in Section 8.5.

The control process of the robots is described in Figure 9.3. The laptop centralises the control. From the measured horizontal positions of Inky and Blinky, the desired positions of the ROVs are computed with the formation controller. Then, a position-tracking controller computes the thruster setpoint sent to Inky and Blinky. In parallel, the depth and attitude of the ROVs are controlled by an independent decoupled controller. Finally, the Buoy uses its USBL and IMU to measure the position of the two ROVs. Note that the ROVs do not communicate or see each other to control their position.

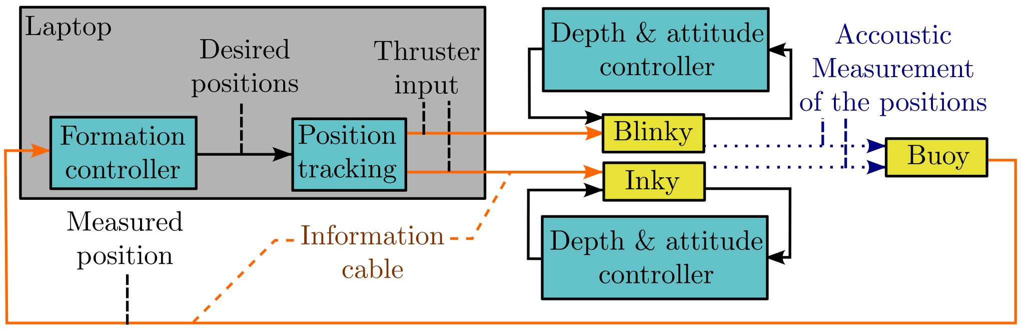


Figure 9.3: Control architecture of the system

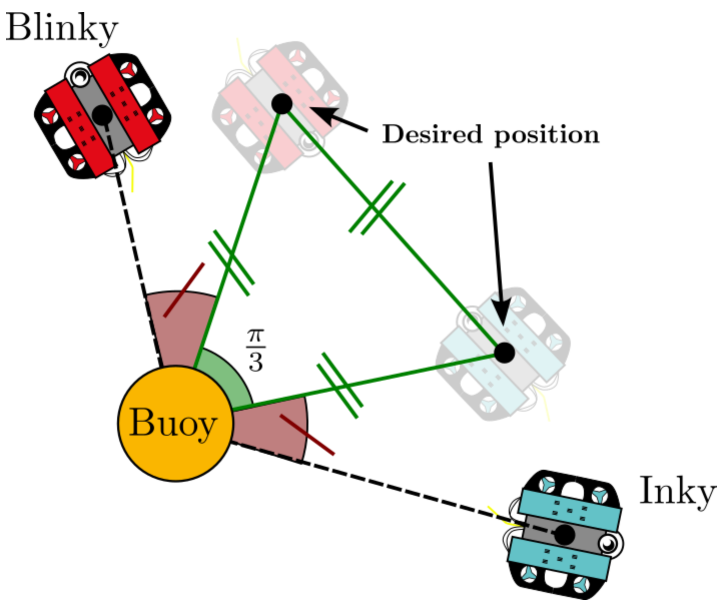


Figure 9.4: Triangular desired formation of the ROVs. The buoy is the third vertices of the equilateral triangle.

Stability analysis For the stability analysis, the ROVs can be described by the synchronous hybrid dynamical system (8.60) of Section 8.5, with the parameters of Table 8.1. This system was proved stable in Section 8.5. Of course, (8.60) is a simple model that approximate the real dynamics of the ROVs, assuming that:

- there is no saturation nor dead zone in the thrusters,
- the buoy measurements are periodic and are not subject to noise,
- the ROVs are only subject to linear damping,
- there is no water current,
- the depth and attitude dynamics are decoupled from the horizontal position dynamics.

Moreover, the controllers of (8.60) are a simplified version of the real controllers.

9.1.3 ROV centralised localisation

The horizontal position of the ROV is computed from the measurements of the USBL and the buoy's IMU, as illustrated by Figure 9.5. The USBL measures the azimuth θ_a, the elevation θ_e and the range r of the ROV in the USBL frame $\{\boldsymbol{e}_{x,\text{buoy}}, \boldsymbol{e}_{y,\text{buoy}}, \boldsymbol{e}_{z,\text{buoy}}\}$. Moreover, the buoy's IMU measures the rotation matrix $\boldsymbol{R}_{\text{buoy}}$ that describes the attitude of the USBL, such that

$$\begin{bmatrix} \boldsymbol{e}^T_{x,\text{world}} \\ \boldsymbol{e}^T_{y,\text{world}} \\ \boldsymbol{e}^T_{z,\text{world}} \end{bmatrix}^T = \boldsymbol{R}_{\text{buoy}} \cdot \begin{bmatrix} \boldsymbol{e}^T_{x,\text{buoy}} \\ \boldsymbol{e}^T_{y,\text{buoy}} \\ \boldsymbol{e}^T_{z,\text{buoy}} \end{bmatrix}^T$$

with the wold frame $\{\boldsymbol{e}_{x,\text{world}}, \boldsymbol{e}_{y,\text{world}}, \boldsymbol{e}_{z,\text{world}}\}$. From these measurements, the horizontal position of the ROV is computed as

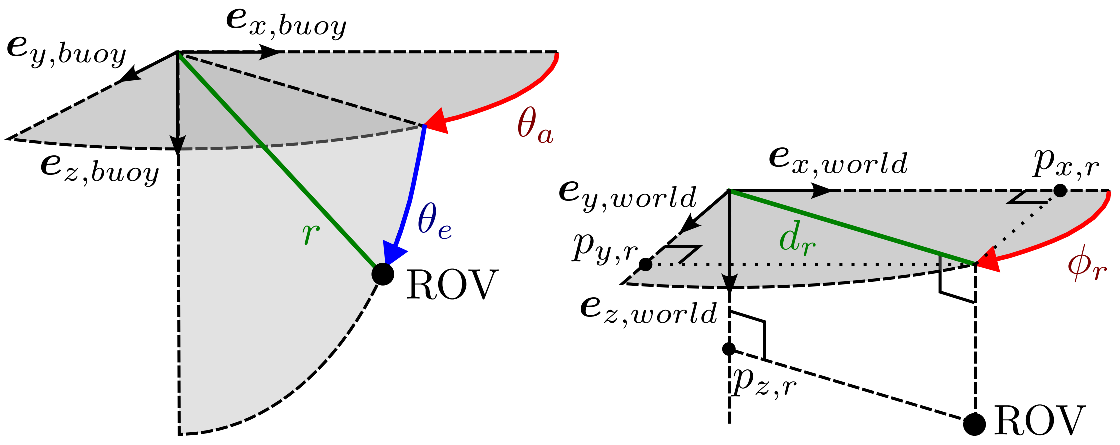
$$\begin{bmatrix} p_{x,r} \\ p_{y,r} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \boldsymbol{R}_{\text{buoy}} \cdot \begin{bmatrix} r \cdot \cos(\theta_a) \cdot \cos(\theta_e) \\ r \cdot \sin(\theta_a) \cdot \cos(\theta_e) \\ r \cdot \sin(\theta_e) \end{bmatrix}, \tag{9.1}$$

and the polar coordinates are then computed as

$$d_r = \sqrt{p_{x,r} + p_{y,r}}, \tag{9.2}$$

$$\phi_r = \text{atan2}(p_{y,r}, p_{x,r}). \tag{9.3}$$

Note that, for depth control, the vertical position of the robot $p_{z,r}$ is measured with the barometer which is more precise than the USBL+IMU.


$\boldsymbol{e}_{x,buoy}$
$\boldsymbol{e}_{y,buoy}$
$\boldsymbol{e}_{z,buoy}$
θ_a
θ_e
r
ROV
$\boldsymbol{e}_{x,world}$
$p_{x,r}$
$\boldsymbol{e}_{y,world}$
$p_{y,r}$
d_r
ϕ_r
$\boldsymbol{e}_{z,world}$
$p_{z,r}$
ROV

(a) USBL measurement (θ_a, θ_e, r) (b) Position of the ROV (d_r, ϕ_r)

Figure 9.5: Localisation of the ROV

9.1.4 Experimental protocol

The experiments presented in this chapter were made during the *Submeeting* event, between the 27th and the 31th of May 2024, at Saint-Raphaël in the south-east of France. The water current is negligible for these tests.

Protocol to evaluate the precision of the localisation To evaluate the period and the precision of the USBL, 8 tests were conducted. One ROV was in the water, with a fixed position, which was measured. From the measurement of this fixed position, the standard deviation of the period and the measurement can be computed to evaluate the precision. Two different environments were tested: the shallow water and the sea. The depth and the radius of the ROVs during these tests are given by Table 9.1.

Tests 1,2,3 and 4 were conducted in shallow water, next to the pontoons of the Port de Boulouris. *Blinky* was placed on the seabed at different distances from the buoy and had a fixed position. The buoy was sheltered from the swell. Since the USBL is in shallow water, there were a lot of acoustic disturbances.

Tests 5,6,7 and 8 were conducted at sea next to Le Lion de Mer at Saint-Raphael. *Inky* was controlled to stay at a fixed depth. The horizontal position was not controlled and was assumed constant. The buoy oscillated with the swell. Since the USBL is at sea, there were fewer acoustic disturbances.

Test number	1	2	3	4	5	6	7	8
Depth (m)	1.6	1.6	1.5	1.7	4.4	4.5	10.5	11
Radius (m)	3.2	6.4	14	1.5	6.8	6.5	7.8	5.6

Table 9.1: Depths of the ROV

Formation control protocol The formation control with two ROVs was tested with the minimum USBL delay ($\simeq 4\,\mathrm{s}$ between two measurements), in different configurations. Test 12 is a formation control conducted in shallow water, at the same place as tests 1 to 4 with the same weather conditions, illustrated by Figure 9.6. Tests 9 to 11 are formation control conducted at sea right after tests 5 to 8.

Protocol to test the limit of stability In addition, to test the limit of the stability of the system with respect to delays, an additional delay δt was added between the USBL measurement at time t and the use of this measurement by the controller at time $t+\delta t$. This artificial delay can simulate a filtering delay in the USBL measurement, which would be required if the acoustic signal were to deteriorate. In theory, when the delays increase, the controller uses older information, and so the stability of the system decreases. However, the measurement period stayed the same.

This delay was applied to the tests 13 to 15. The additional delay is 1 s for test 13, 2 s for test 14 and 4 s for test 15. Tests 13 to 15 were conducted in shallow water, right after test 12.

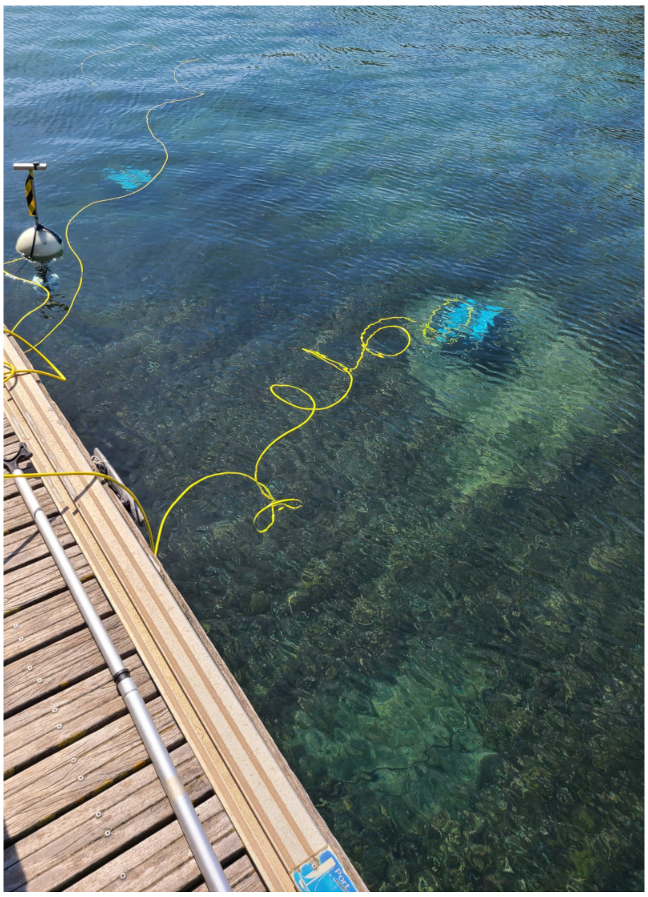


Figure 9.6: Formation control tests at the Port de Boulouris, Saint-Raphaël, France

9.2 Results

9.2.1 The DataSet

The data of the ROVs and the buoy were recorded and stored in rosbag files. It is possible to replay the experiments with the ROS middleware. Moreover, with the RViz software, a 3d replay of the experiments can be visualised as illustrated by Figure 9.7. A video and more information about the formation control can be found on morgan-louedec.fr/submeeting-2024/ .

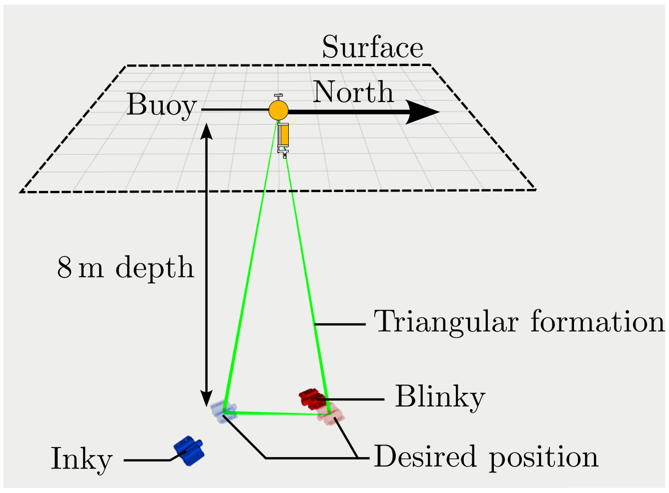


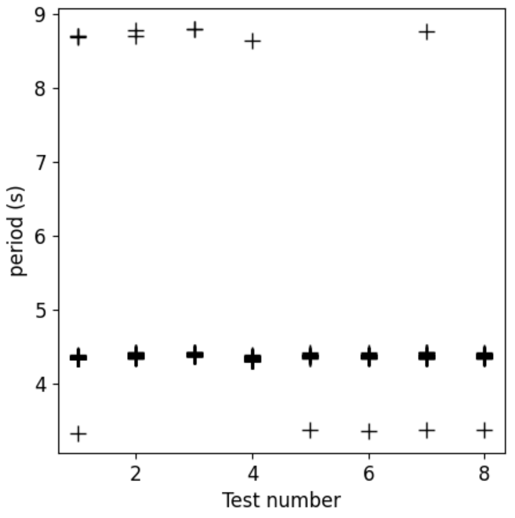
Figure 9.7: 3d reconstruction of the test 11

9.2.2 The quality of the localisation

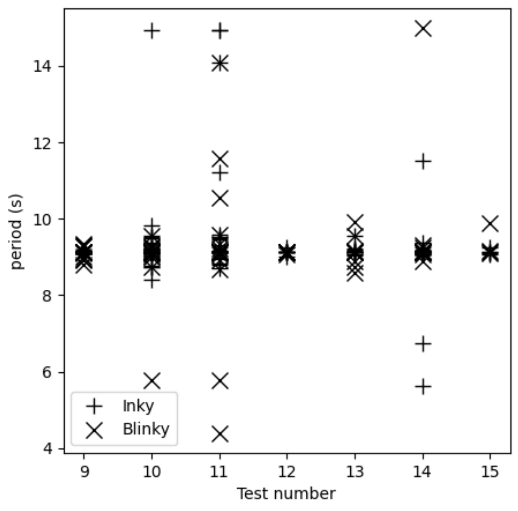
As presented in Section 9.1.4, tests 1 to 8 were conducted to evaluate the quality of localisation. In this section, the quality is evaluated by computing standard deviations for the USBL measurement period and the measured ROV position.

Measurement period Figure 9.8a presents an evaluation of the USBL measurement period T for the tests 1 to 8, where only one ROV is localised. There is a marker for each measured period. There is 3% of outliers. Some outliers correspond to $2 \cdot T$ when a measurement is missed because the USBL is not able to receive the response from the ROV. Without the outliers, the mean period is $T = 4.37\,\text{s}$ with a standard deviation of $\sigma_T = 0.03\,\text{s}$.

Figure 9.8b presents the same evaluation for tests 9 to 15, where two ROVs are localised. There is 7% of outliers. Without the outliers, the mean period is $T = 9.14\,\text{s}$ with a standard deviation of $\sigma_T = 0.17\,\text{s}$. As expected, with two ROVs in the water, the USBL must alternate the measurements, so the period is twice as long. Moreover, with two ROVs, the measurements are less regular. Nevertheless, the standard deviation is slow enough to assume a constant measurement period.



(a) With one ROV.



(b) With two ROVs

Figure 9.8: Measurement period of the USBL. There is a marker for each measurement period. One can observe an average period of 4.37 s for one ROV, and 9.14 s for two ROVs.

Position measurement The USBL measurements are displayed on Figure 9.10, the IMU measurements are displayed on Figure 9.9, the estimations of the ROV polar coordinates (d_r, ϕ_r) are displayed on Figure 9.11 and the standard deviation of the localisation (σ_d, σ_ϕ) is displayed on Figure 9.12. Since the position of the ROV is assumed fixed, the oscillations in Figure 9.11 are considered estimation errors.

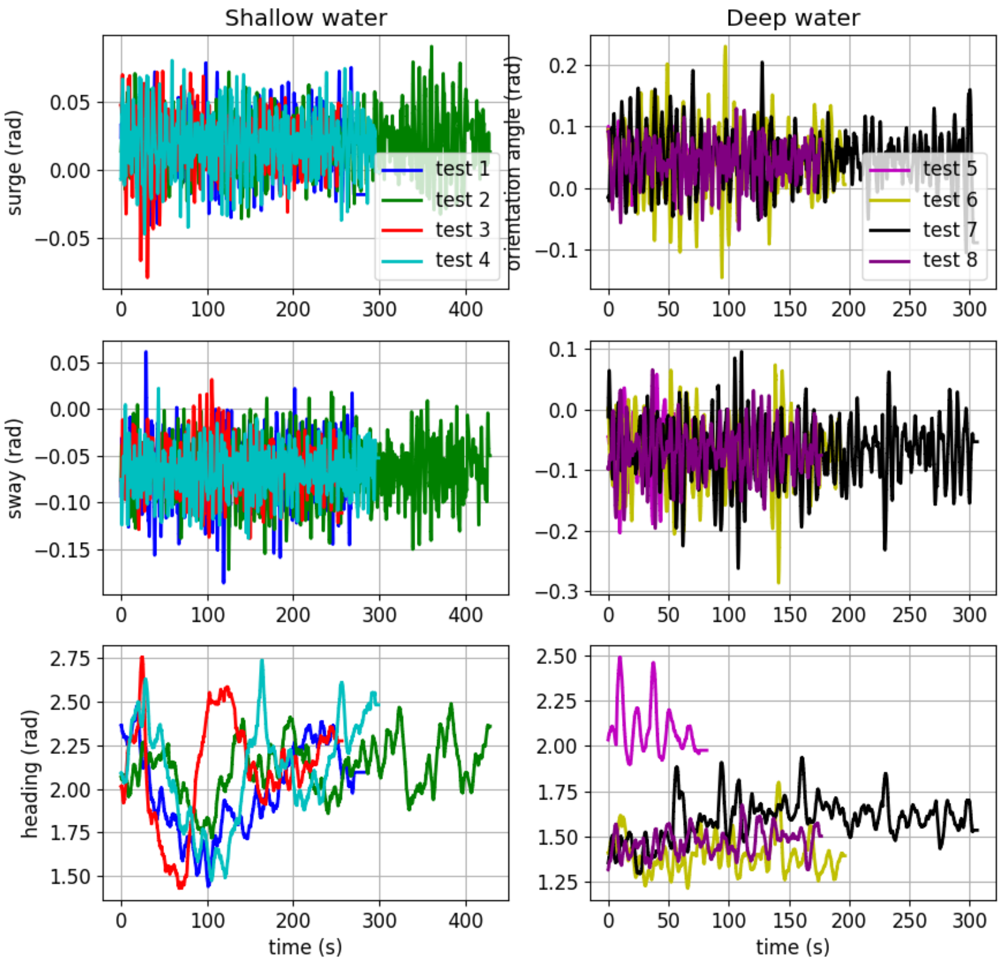


Figure 9.9: Buoy IMU measurement for the tests 1 to 8

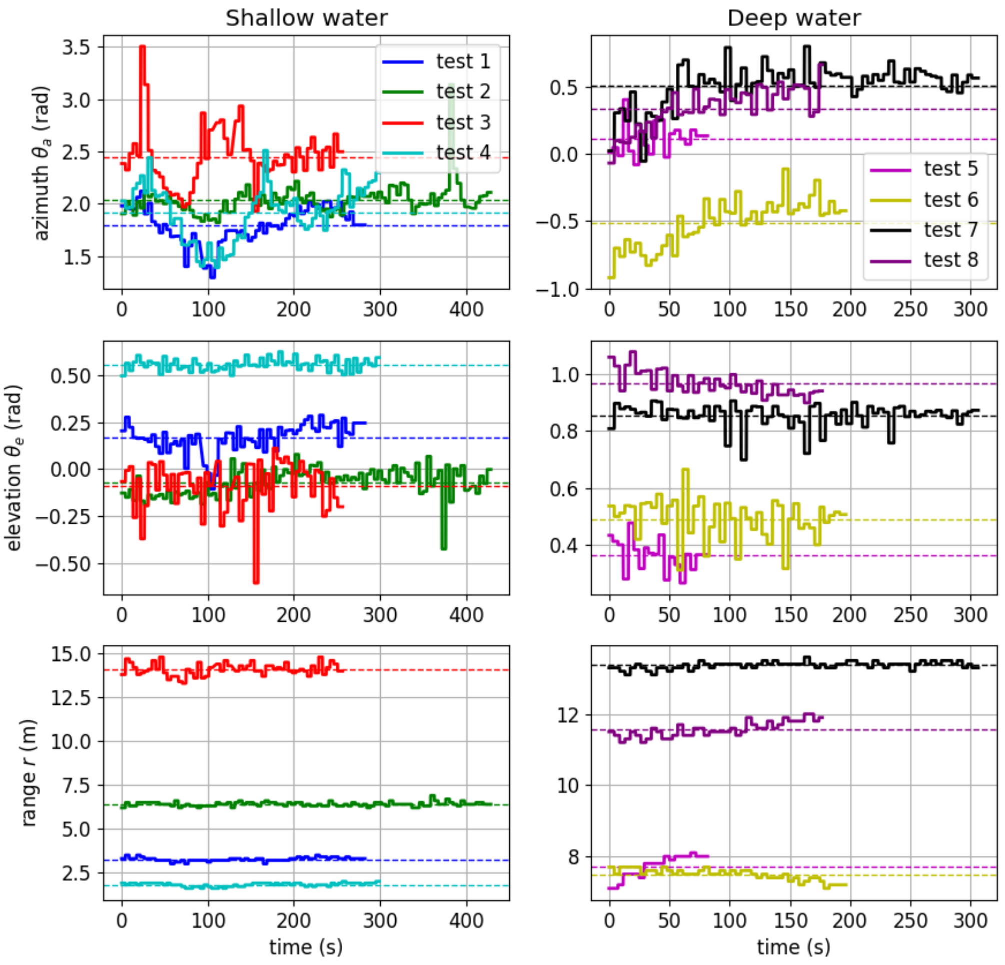


Figure 9.10: USBL measurements for the tests 1 to 8

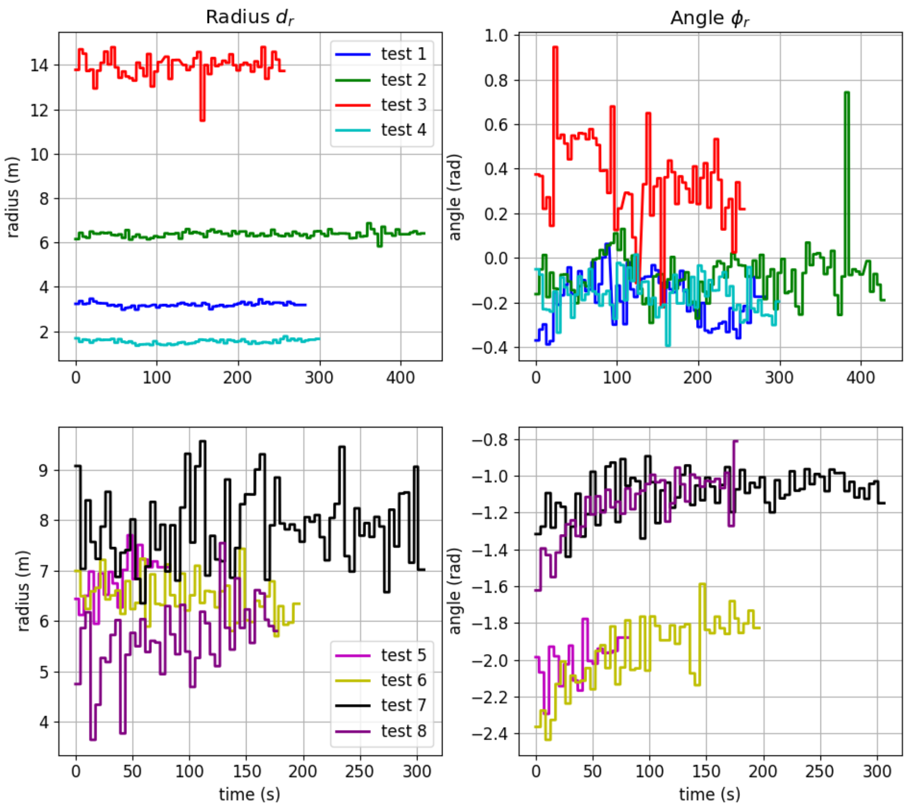


Figure 9.11: Measurement of the position (d_r, ϕ_r) for the tests 1 to 8

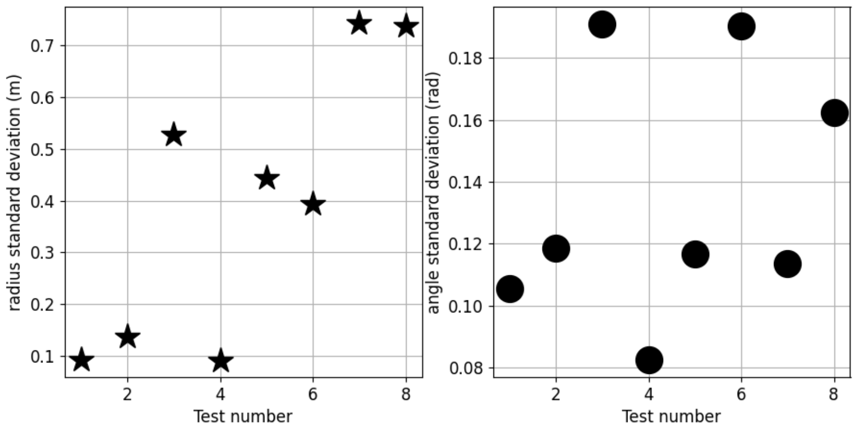


Figure 9.12: Standard deviation of the position (d_r, ϕ_r) with respect to the test number

As visible in Figure 9.9, the sway and the surge of the buoy oscillate with the swell. The amplitude is about 0.05 rad in shallow water (tests 1 to 4) and 0.1 rad at sea (tests 5 to 8). Thus, the swell is more important at sea. Moreover, the heading of the buoy changes over time in a range of ± 0.5 rad. Therefore, it is important to compensate for the buoy motion in the localisation of the ROV.

In addition, on Figure 9.12, the radius d_r is more precise in shallow water, for the tests 1,2 and 4 ($\sigma_d < 0.2$ m), except for the test 3 ($\sigma_d \simeq 0.52$ m). Moreover, at sea, σ_d is higher for the deep tests 7 and 8 ($\sigma_d > 0.7$ m) compared to the less deep tests 5 and 6. Regarding the angle ϕ_r, the standard deviation is higher for the tests 3, 7 and 8 ($\sigma_d > 0.16$ rad) compared to the other tests ($\sigma_d < 0.12$ rad). All these differences can be explained as follows.

The test 3 is the worst configuration for the USBL, acoustically speaking. In the test 1 to 4, the ROV is in shallow water, at a depth of about 1.5 m. The motion of the buoy is similar for these four tests, as illustrated by Figure 9.9. While for tests 1,2 and 4, the ROV is under a radius of 7 m, the radius is about 14 m for test 3. As a result, there are more acoustic disturbances for test 3. Due to the shallow water, there are more reflections of the acoustic signal between the surface and the seafloor. These reflections create more acoustic multi-path which reduces the precision of the USBL measurement. Thus the quality of the USBL measurement is deteriorated for test 3, as illustrated by Figure 9.10. For the tests 1,2 and 4, the standard deviation of θ_e is about 0.07 rad and the standard deviation of the range r is about 0.1 m. However, for test 3, their standard deviations are respectively 0.13 rad and 0.3 m. In practice, although the X150 micro-USBL is presented as omnidirectional on the datasheet of the manufacturer, the USBL localisation is often accurate when robots are inside a cone under the buoy. The omnidirectionality of the USBL has not been tested. For test 3, the ROV is too probably far from this cone to be detected accurately.

Moreover, the measurement of the buoy's IMU and the buoy's USBL were not precisely synchronised. As a result, when the buoy oscillates with the swell, the amplitude of the localisation error is increased. In Figure 9.9, the buoy oscillates more at sea than in shallow water. This can explain why the sea tests (5 to 8) have a higher σ_s, even though there is less acoustic disturbance at sea. Furthermore, this synchronisation error creates an error of angle, so the radius error is proportional to the radius. This is why σ_d is higher for tests 7 and 8 at a depth of about 10 m compared to the test 5 and 6 at a depth of about 5 m. Because of this synchronisation error, the localisation is not reliable in heavy swell.

In addition, on Figure 9.11, for the tests 6 and 8, the angle ϕ_r drifts. The ROV probably moved during these two tests. So the value of σ_ϕ is not relevant for tests 6 and 8.

As a result, the error of precision of the localisation of the ROV is non-neglectable and has some impact on the formation control. From Figure 9.12, the standard deviation of the radius d_r and the angle ϕ_r can be evaluated as

$$\begin{aligned} \sigma_d &\simeq 0.4\,\text{m}, \\ \sigma_\phi &\simeq 0.12\,\text{rad}. \end{aligned} \tag{9.4}$$

9.2.3 The quality of the formation control

From the data of the experiments, it is possible to display the discrete state vector

$$\boldsymbol{m}_j = \left[\begin{array}{c} d_{b,j} - d^*, \\ d_{r,j} - d^*, \\ \phi_{r,j} - \phi_{b,j} - \frac{\pi}{3}. \end{array} \right] \tag{9.5}$$

of the synchronous hybrid system presented in Section 8.41. The state of the system is expected to converge in a neighbourhood of the equilibrium point, whose size is of the same order as σ_d for $m_{1,j}$ and $m_{2,j}$ and the same order as $2 \cdot \sigma_\phi$ for $m_{3,j}$.

Figure 9.13 presents the time evolution of $\boldsymbol{m}_k$ for the tests 9, 10, 11 and 12. During these tests, the system was stable and the ROVs converged towards a neighbourhood of the equilibrium point. The size of the neighbourhood is visually coherent with the standard deviation of the ROV localisation. In this neighbourhood, the ROVs can see each other as illustrated by Figure 9.14. The videos of the formation control can be found on morgan-louedec.fr/submeeting-2024/ .

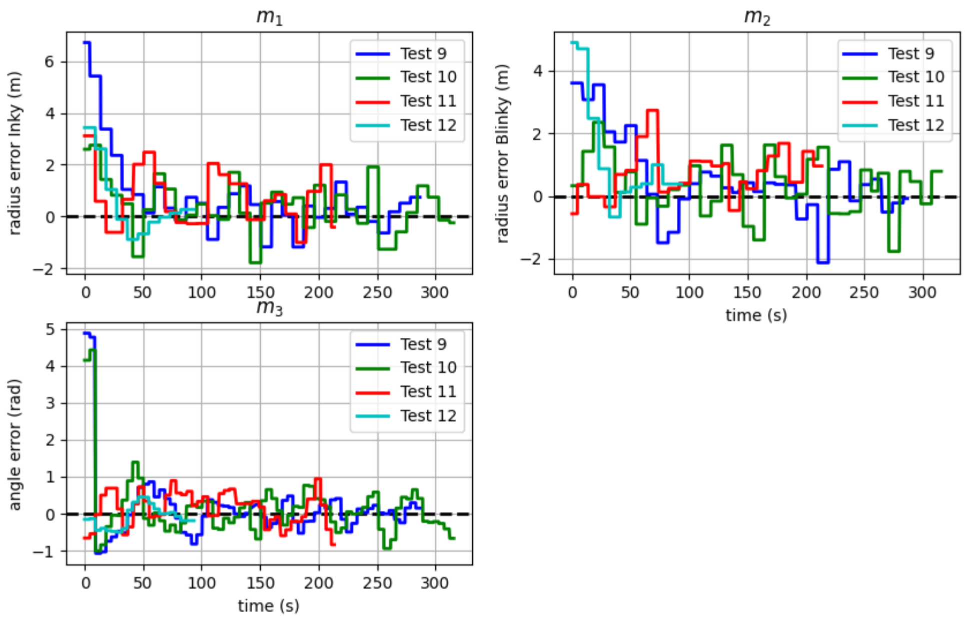
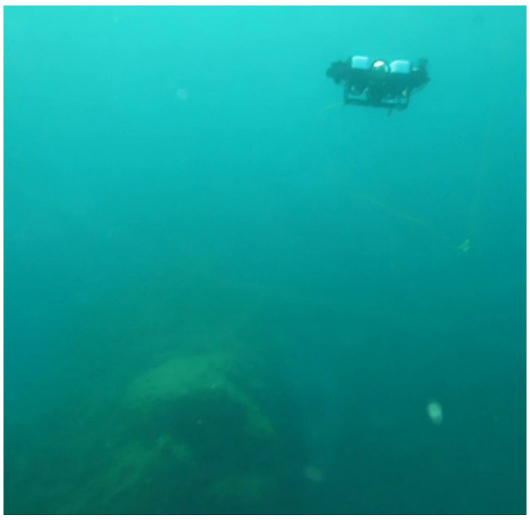


Figure 9.13: Results of the formation control.


(a) Test 10 at sea

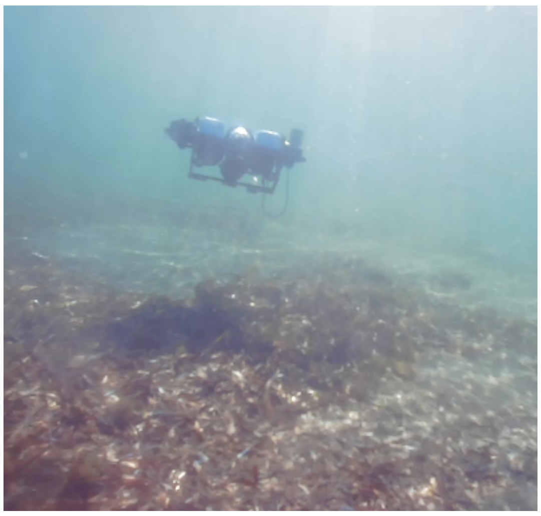

(b) Test 12 in Shallow water

Figure 9.14: Inky sees Blinky

Deterioration of the stability with additional delays. Figure 9.15 presents the time evolution of $\boldsymbol{m}_k$ for the tests 13, 14 and 15, where a delay δt is added in the USBL measurement. the USBL measurement at time t is used by the controller at time $t + \delta t$. The additional delay is 1 s for test 13, 2 s for test 14 and 4 s for test 15.

One can observe that the formation converges to a stable neighbourhood with 1 s and 2 s of additional delays. However, the state m_3 starts to diverge for the test 15 with 4 s of delay, hence the premature stop of this test. During test 15, the ROVs spun around the buoy, one ROV was fleeing, and the second ROV was chasing the first.

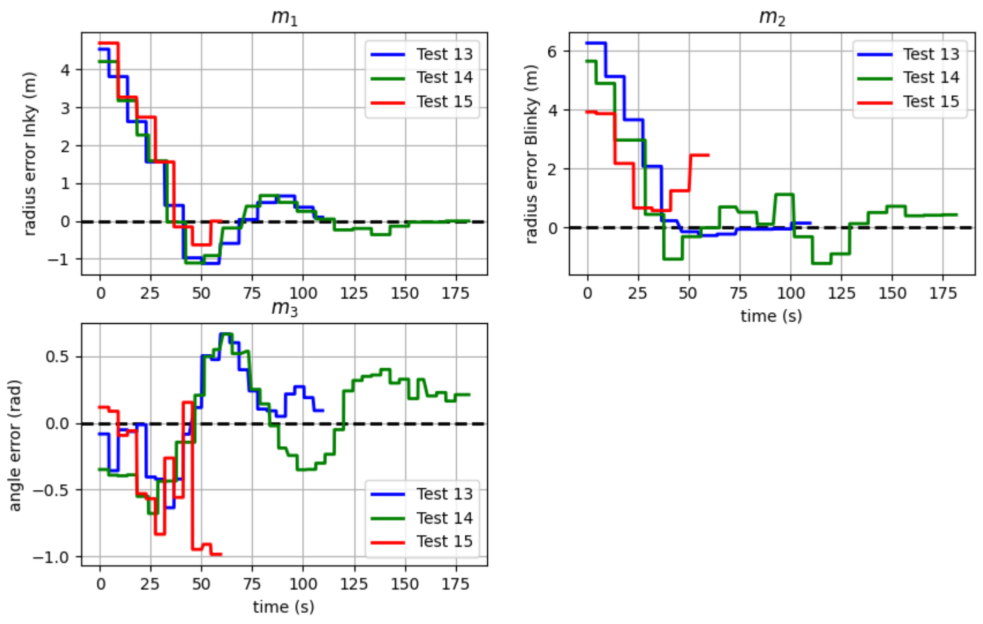


Figure 9.15: Formation control with additional delay in the USBL measurement.

9.3 Discussion

Following the results of Section 9.2, the triangular formation control with two ROVs is achieved with two ROVs. The control is stable in the sense that the robots converge in a neighbourhood of the equilibrium point. The size of this neighbourhood is coherent with the precision of the localisation.

Modelling the system with a hybrid system is relevant because the minimum USBL measurement period is above 4 s. Thus, compared to the continuous dynamic of the ROVs, the USBL measurements must be considered as periodic discrete-time events.

Moreover, the acoustic localisation is harder to achieve for a large number of robots. The USBL of the experiments locates only one robot at a time with a ping. Thus, the USBL measurement period proportionally increases with the number of robots ($+4\,s$ per robot). Sending pings to different robots at the same time can reduce the period, but the precision of the localisation is deteriorated by ping interference. An alternative is to synchronise the buoy and the robots with the same clock, so that the buoy simply listens to the acoustic messages sent by the robots, knowing in advance the time of their transmission. But it's not easy to synchronise all the robots.

In addition, the experiments show an interest in computing a positive invariant sets when the dynamical system has bounded disturbances. Because of the acoustic disturbances, USBL localisation precision is limited to a decimetre. To enhance the stability analysis of the system, a localisation error should be considered in the model. This error should be bounded by the localisation precision. Adapting all the numerical methods of this thesis to include bounded disturbances will be subject to future work. Some preliminary results are given in [62].

However, the ellipsoid computed in the stability analysis is smaller than the experimental stable neighbourhood, because of the pessimism in the computations (a scale 0.01 m of compared to 0.1 m). Although the numerical methods developed in this thesis can compute high-dimensional positive invariant sets, it can only compute small sets.

Moreover, the ground truth was not recorded during the experiments. Future experiments should compare the discrete-time measured positions with the real continuous-time positions of the robots in order to show that the continuous part of the system is also stable. However, to record the real position of the robots, one needs a high-precision and high-frequency measurement of the position, which is difficult to implement in an outdoor environment.

In addition, the localisation precision deteriorated because the buoy's IMU and USBL were not precisely synchronised. As the buoy moves with the swell, it creates some error in the computation of the position in the world frame. However, if the IMU and the USBL are precisely synchronised, the localisation is not deteriorated by the buoy's movements, especially when the robots are far from the buoy.

Finally, as expected, adding delays in the control loop decreases the stability of the formation control. Therefore, delays must be considered in the stability analysis, when they are significant, to know what is the maximum delay the system can have while remaining stable.

9.4 Conclusion

This chapter presents a real formation control with two ROVs. The theoretical stability of the formation control was validated before the experiment, in Chapter 8. As a result of the experiments, the formation is achievable and stable in practice. However, because of the imprecise localisation, the system only converges in a neighbourhood of the equilibrium point. This neighbourhood is larger than the ellipsoid of stability computed for this system in Chapter 8.

Moreover, although the localisation and the control of the ROVs are centralised, it is possible to make them decentralised. A decentralised implementation would be more efficient with many robots compared to the centralised implementation, as presented in Chapter 2.

In addition, the experiments show the interest of using hybrid systems to model underwater robots, because of the low frequency of the USBL measurements. Moreover, the experiments show the interest of computing a positive invariant set when the dynamical system has bounded disturbances. The localisation precision could be enhanced with a tight synchronisation between the USBL and IMU of the buoy. Future experiments should compare the measured positions with the ground truth. They should also include more robots in the formation.

Chapter 10

Conclusions and Perspectives

Conclusion

In this thesis, a stability analysis method with computer-assisted proof has been proposed to study high-dimensional nonlinear systems. With this method, one can prove the stability of the system without manually computing Lyapunov functions. This method can compute positive invariant ellipsoids and can prove that the system is exponentially stable in this ellipsoid. The numerical computations of the method are guaranteed by the interval algebra. This method can thus be applied to problems of formation control with underwater robots.

The main contributions of this thesis are the following:

- First, the stability analysis method was been designed for high-dimensional nonlinear discrete-time systems. An ellipsoid that is likely positive invariant is computed with the discrete-time axis-aligned Lyapunov equation. Then the ellipsoid is propagated with one step of the recursive mapping of the system, using the guaranteed propagation of ellipsoids from [95]. Finally, the stability is deduced when the result of the propagation is strictly included in the original ellipsoid. This contribution was presented in Chapter 5. Under review at the IEEE Transaction on Automatic Control journal.

- Second, the method is adapted for high-dimensional nonlinear continuous-time systems. To prove the exponential stability, the discrete-time method is applied to a rigorous discretisation of the continuous system. To prove the positive invariance, the method is applied on a discretisation of the system with an Euler scheme. This contribution was presented in Chapter 6 and published in the Automatica journal [62].

- Third, the guaranteed propagation method is generalised to include singular mappings and degenerate ellipsoids. This generalisation is made by adding non-zero eigenvalues, in the singular case. This contribution was presented in Chapter 7 and at the 7th IFAC ACNDC conference [63].

- Fourth, the stability analysis method is adapted for high-dimensional nonlinear synchronous hybrid nonlinear systems. The exponential stability is proved by applying the discrete method on a rigorous discretisation of the system. This contribution was presented in Chapter 8.

- Fifth, real-world experiments illustrate the robustness of formation control with two ROVs. These experiments show that even with an imprecise localisation, the formation can converge to a neighbourhood of the equilibrium point. This contribution was presented in Chapter 9.

The method supposes that the mappings of the systems are Lipschitz. When the system has a continuous-time variable, the method evaluates the Jacobian of the flow using a guaranteed integration of the variational equation.

This method is unusual in the science domain of automation as it relies on tools such as guaranteed integration algorithms. However, it is related to Lyapunov Theory as the ellipsoids involved in the method defined levels of quadratic Lyapunov functions. Compared to a manual search for a Lyapunov function, the method is easier to apply to complex systems with the presence of non-linearity, high dimensions and interactions between continuous and discrete variables. It has the potential to solve stability problems when Lyapunov functions have not been found yet.

However, the method is subject to pessimism in the ellipsoidal propagation. Thus, in practice, the computed ellipsoids are sometimes very small. The pessimism increases when the system becomes more complex (with higher dimensions, more non-linearity, more intermediate discrete events,...) and when the system is less stable. Furthermore, some recommendations for the implementation of the method are listed below:

- Pessimism can be limited with a good conditioning of the system. The system must be simplified by removing redundant variables and by normalisation.

- When the ellipsoids are almost degenerated there is a border effect which results in a lot of pessimism. Some additional manual tuning is required to reduce the pessimism.

- If the system is barely stable, high-precision libraries are needed to handle small ellipsoids and small pessimism. Moreover, with better precision, bigger positive invariant ellipsoids may be found.

Perspective

Several midterm and long-term directions are proposed below. First, the ellipsoidal stability analysis method can be extended to asynchronous hybrid systems. These systems are more complex to study, considering that the time of the discrete updates can be state-dependent and change with the initial condition. To deal with the variation of the discrete update time, the hybrid system can be discretised with a Poincare method, presented in [35]. With this complex discretisation, the system is described by mappings, called Poincare mappings, defined on sections of $\mathbb{R}^n$ that are transverse to the flow of the continuous dynamics, as illustrated by Figure. There exists some numerical method to compute the Jacobian of a Poincare mapping, as in [42]. Thus It is possible to adapt the stability analysis method to a Poincare discretisation, in order to study an asynchronous hybrid system.

Secondly, the ellipsoidal stability analysis method can be extended to systems with bounded perturbations. As shown in Chapter 9, underwater robots are subject to non-neglectable perturbations. Assuming their presence, the robots only converge on a neighbourhood of the formation. The ellipsoidal stability method could characterise this neighbourhood by computing a positive invariant ellipsoid. To extend the method to perturbations, the guaranteed ellipsoidal propagation method must also be extended to systems with bounded perturbation.

Thirdly, the implementation of the ellipsoidal stability analysis could be more rigorous. The interval operations in the algorithms of this thesis have not all been detailed. The applications of the thesis use some operations that are not fully rigorous or time-efficient. By developing a library of efficient interval operations for ellipsoids, the pessimism in the stability analysis could be reduced.

Finally, additional experiments could be implemented with more robots and more complex collaboration tasks. With a large fleet, additional challenges will appear and the formation stability will be more challenging to obtain.

Bibliography

[1] Matthias Althoff and Jagat Jyoti Rath. Comparison of guaranteed state estimators for linear time-invariant systems. *Automatica*, 130:109662, August 2021.

[2] Vahid Aryai, Rouzbeh Abbassi, and Nagi Abdussamie. Reliability of multipurpose offshore-facilities: Present status and future direction in Australia. *Process Safety and Environmental Protection*, 148:437–461, April 2021.

[3] T. Balch and R.C. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939, December 1998. Conference Name: IEEE Transactions on Robotics and Automation.

[4] Yasmina Becis-Aubry. Ellipsoidal constrained state estimation in presence of bounded disturbances. In *2021 European Control Conference (ECC)*, pages 555–560, June 2021.

[5] Martin Berz and Kyoko Makino. New methods for high-dimensional verified quadrature. *Reliable Computing*, 5(1):13–22, 1999. ISBN: 1385-3139 Publisher: Springer.

[6] Franco Blanchini and Stefano Miani. *Set-Theoretic Methods in Control*. Systems & Control: Foundations & Applications. Springer International Publishing, Cham, 2015.

[7] Nicoletta Bof, Ruggero Carli, and Luca Schenato. Lyapunov Theory for Discrete Time Systems. Technical report, arXiv, 2018.

[8] Auguste Bourgois. *Safe & collaborative autonomous underwater docking: interval methods for proving the feasibility of an underwater docking problem*. PhD thesis, ENSTA Bretagne-École nationale supérieure de techniques avancées Bretagne, 2021.

[9] Auguste Bourgois and Luc Jaulin. Interval centred form for proving stability of non-linear discrete-time systems. *Electronic Proceedings in Theoretical Computer Science*, 331:1–17, January 2021.

[10] Auguste Bourgois, Simon Rohou, Luc Jaulin, and Andreas Rauh. Proving Feasibility of a Docking Mission: A Contractor Programming Approach. *Mathematics*, 10(7):1130, January 2022. Number: 7 Publisher: Multidisciplinary Digital Publishing Institute.

[11] S. Brown. Families of Robotic Systems Solving Offshore Wind Inspection Challenges. OnePetro, April 2024.

[12] Francesco Bullo. *Contraction Theory for Dynamical Systems.* Kindle Direct Publishing, 1.1 edition, 2023.

[13] Khalid Manzoor Butt and Sadaf Jan Siddiqui. Growing Chinese Presence in the Indian Ocean: Prospects and Challenges. *Strategic Studies*, 41(2):64–81, 2021. Publisher: Institute of Strategic Studies Islamabad.

[14] Yongcan Cao, Wenwu Yu, W. Ren, and G. Chen. An Overview of Recent Progress in the Study of Distributed Multi-Agent Coordination. *IEEE Transactions on Industrial Informatics*, 2013.

[15] Alexandre Chapoutot. Validated Explicit and Implicit Runge-Kutta Methods. *Reliable Computing*, 2016.

[16] Jian Chen, Dong Sun, Jie Yang, and Haoyao Chen. Leader-Follower Formation Control of Multiple Non-holonomic Mobile Robots Incorporating a Receding-horizon Scheme. *The International Journal of Robotics Research*, 29(6):727–747, May 2010. Publisher: SAGE Publications Ltd STM.

[17] Luca Consolini, Fabio Morbidi, Domenico Prattichizzo, and Mario Tosques. Leader–follower formation control of nonholonomic mobile robots with input constraints. *Automatica*, 44(5):1343–1349, May 2008.

[18] Andrea E. Copping and Lenaig G. Hemery. OES-Environmental 2020 State of the Science Report: Environmental Effects of Marine Renewable Energy Development Around the World. Report for Ocean Energy Systems (OES). Technical Report PNNL-29976, Pacific Northwest National Lab. (PNNL), Richland, WA (United States), September 2020.

[19] Christopher Costello, Ling Cao, Stefan Gelcich, Miguel Á Cisneros-Mata, Christopher M. Free, Halley E. Froehlich, Christopher D. Golden, Gakushi Ishimura, Jason Maier, Ilan Macadam-Somer, Tracey Mangin, Michael C. Melnychuk, Masanori Miyahara, Carryn L. de Moor, Rosamond Naylor, Linda Nøstbakken, Elena Ojea, Erin O'Reilly, Ana M. Parma, Andrew J. Plantinga, Shakuntala H. Thilsted, and Jane Lubchenco. The future of food from the sea. *Nature*, 588(7836):95–100, December 2020. Publisher: Nature Publishing Group.

[20] Julien Damers. *Lie groups applied to localisation of mobile robots.* PhD thesis, Brest, École nationale supérieure de techniques avancées Bretagne, 2022.

[21] A.K. Das, R. Fierro, V. Kumar, J.P. Ostrowski, J. Spletzer, and C.J. Taylor. A vision-based formation control framework. *IEEE Transactions on Robotics and Automation*, 18(5):813–825, October 2002. Conference Name: IEEE Transactions on Robotics and Automation.

[22] Celso De La Cruz and Ricardo Carelli. Dynamic Modeling and Centralized Formation Control of Mobile Robots. In *IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics*, pages 3880–3885, November 2006. ISSN: 1553-572X.

[23] Dimos V. Dimarogonas and Kostas J. Kyriakopoulos. On the Rendezvous Problem for Multiple Nonholonomic Agents. *IEEE Transactions on Automatic Control*, 52(5):916–922, May 2007. Conference Name: IEEE Transactions on Automatic Control.

[24] Marco Dorigo, Guy Theraulaz, and Vito Trianni. Reflections on the future of swarm robotics. *Science Robotics*, 5(49), December 2020. Publisher: American Association for the Advancement of Science.

[25] Andreas Eggers, Nacim Ramdani, Nedialko S. Nedialkov, and Martin Fränzle. Improving the SAT modulo ODE approach to hybrid systems analysis by combining different enclosure methods. *Software & Systems Modeling*, 14(1):121–148, February 2015.

[26] Pieter Eijgenraam. The solution of initial value problems using interval arithmetic: formulation and analysis of an algorithm. *MC Tracts*, 1981. Publisher: Centrum Voor Wiskunde en Informatica.

[27] Willem Esterhuizen, Tim Aschenbruck, and Stefan Streif. On maximal robust positively invariant sets in constrained nonlinear systems. *Automatica*, 119:109044, September 2020.

[28] Z. Gajic and M Qureshi. *Lyapunov matrix equation in system stability and control*. Courier corporation edition, 2008.

[29] Antoine Girard. *Analyse Algorithmique des Systèmes Hybrides*. phdthesis, Institut National Polytechnique de Grenoble - INPG, September 2004.

[30] Rafal Goebel, Ricardo G Sanfelice, and Andrew R Teel. Hybrid dynamical systems. 2009.

[31] Eric Goubault and Sylvie Putot. A zonotopic framework for functional abstractions. *Formal Methods in System Design*, 47(3):302–360, December 2015.

[32] Camila M. G. Gussen, Paulo S. R. Diniz, Marcello L. R. Campos, Wallace A. Martins, Felipe M. Costa, and Jonathan N. Gois. A Survey of Underwater Wireless Communication Technologies. *Journal of Communication and Information Systems*, 31(1), October 2016. Number: 1.

[33] Behnaz Hadi, Alireza Khosravi, and Pouria Sarhadi. A Review of the Path Planning and Formation Control for Multiple Autonomous Underwater Vehicles. *Journal of Intelligent & Robotic Systems*, 101(4):67, March 2021.

[34] James D. Hamilton. *Time Series Analysis.* Princeton University Press, princeton university press edition, 2020.

[35] M. Henon. On the numerical computation of Poincaré maps. *Physica D: Nonlinear Phenomena*, 5(2):412–414, September 1982.

[36] Laurentiu Hetel, Christophe Fiter, Hassan Omran, Alexandre Seuret, Emilia Fridman, Jean-Pierre Richard, and Silviu Iulian Niculescu. Recent developments on the stability of systems with aperiodic sampling: An overview. *Automatica*, 76:309–335, February 2017.

[37] Morris W. Hirsch, Stephen Smale, and Robert L. Devaney. *Differential Equations, Dynamical Systems, and an Introduction to Chaos.* Elsevier, 2013.

[38] Andrea Iannelli, Andrés Marcos, and Mark Lowenberg. Robust estimations of the Region of Attraction using invariant sets. *Journal of the Franklin Institute*, 356(8):4622–4647, May 2019.

[39] Luc Jaulin, Michel Kieffer, Olivier Didrit, and Eric Walter. *Applied Interval Analysis.* Springer, London, 2001.

[40] Bibek Kabi. *Synthesizing invariants : a constraint programming approach based on zonotopic abstraction.* phdthesis, Institut Polytechnique de Paris, June 2020.

[41] Tomasz Kapela, Marian Mrozek, Daniel Wilczak, and Piotr Zgliczynski. CAPD::DynSys: A flexible C++ toolbox for rigorous numerical analysis of dynamical systems. *Communications in Nonlinear Science and Numerical Simulation*, 101:105578, October 2021.

[42] Tomasz Kapela, Daniel Wilczak, and Piotr Zgliczynski. Recent advances in a rigorous computation of Poincaré maps. *Communications in Nonlinear Science and Numerical Simulation*, 110:106366, July 2022.

[43] Tomasz Kapela and Piotr Zgliczynski. A Lohner-type algorithm for control systems and ordinary differential inclusions. Technical Report arXiv:0712.0910, arXiv, December 2007.

[44] Hassan K. Khalil. *Nonlinear Systems.* Pearson, Upper Saddle River, NJ, 3rd edition, December 2001.

[45] Suleman Khan, Irshad Hussain, and Muhammad Irfan Khattak. Consensus Based Formation Control of Multiple UAVs. *Journal of Information Communication Technologies and Robotic Applications*, pages 31–37, June 2020.

[46] Michel Kieffer, Luc Jaulin, Isabelle Braems, and Eric Walter. Guaranteed set computation with subpavings. 2000.

[47] Shreyas Kousik, Adam Dai, and Grace Gao. Ellipsotopes: Combining Ellipsoids and Zonotopes for Reachability Analysis and Fault Detection. Technical Report arXiv:2108.01750, arXiv, June 2022.

[48] Shreyas Kousik, Adam Dai, and Grace X. Gao. Ellipsotopes: Uniting Ellipsoids and Zonotopes for Reachability Analysis and Fault Detection. *IEEE Transactions on Automatic Control*, pages 1–13, 2022. Conference Name: IEEE Transactions on Automatic Control.

[49] F. Kruckeberg. *Ordinary Differential Equations, Topics in Interval Analysis, ed. E. Hansen.* Clarendron Press, Oxford, 1969.

[50] Alex A. Kurzhanskiy and Pravin Varaiya. Ellipsoidal toolbox (ET). In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 1498–1503. IEEE, 2006.

[51] Alex A. Kurzhanskiy and Pravin Varaiya. Theory and computational techniques for analysis of discrete-time control systems with disturbances. *Optimization Methods and Software*, 26(4-5):719–746, October 2011.

[52] Thomas Le Mezo, Luc Jaulin, and Benoit Zerr. An Interval Approach to Compute Invariant Sets. *IEEE Transactions on Automatic Control*, 62(8):4236–4242, August 2017. Conference Name: IEEE Transactions on Automatic Control.

[53] Giroung Lee and Dongkyoung Chwa. Decentralized behavior-based formation control of multiple robots considering obstacle avoidance. *Intelligent Service Robotics*, 11(1):127–138, January 2018.

[54] Min Cheol Lee and Min Gyu Park. Artificial potential field based path planning for mobile robots using a virtual obstacle concept. In *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, volume 2, pages 735–740 vol.2, July 2003.

[55] M. Anthony Lewis and Kar-Han Tan. High Precision Formation Control of Mobile Robots Using Virtual Structures. *Autonomous Robots*, 4(4):387–403, October 1997.

[56] Qi Li, Jinyuan Wei, Qiuxiong Gou, and Zhiqi Niu. Distributed adaptive fixed-time formation control for second-order multi-agent systems with collision avoidance. *Information Sciences*, 564:27–44, July 2021.

[57] Xin Li, Daqi Zhu, and Yuang Qian. A Survey on Formation Control Algorithms for Multi-AUV System. *Unmanned Systems*, 02(04):351–359, October 2014. Publisher: World Scientific Publishing Co.

[58] Jie Lian and Feiyue Wu. Stabilization of Switched Linear Systems Subject to Actuator Saturation via Invariant Semiellipsoids. *IEEE Transactions on Automatic Control*, 65(10):4332–4339, October 2020. Conference Name: IEEE Transactions on Automatic Control.

[59] Zenghui Liu, Kai Liu, Xuguang Chen, Zhengkuo Ma, Rui Lv, Changyun Wei, and Ke Ma. Deep-sea rock mechanics and mining technology: State of the art and perspectives. *International Journal of Mining Science and Technology*, 33(9):1083–1115, September 2023.

[60] Chengqi Long, Manjiang Hu, Xiaohui Qin, and Yougang Bian. Hierarchical trajectory tracking control for ROVs subject to disturbances and parametric uncertainties. *Ocean Engineering*, 266:112733, December 2022.

[61] Morgan Louédec, L Jaulin, and Christophe Viel. Outer enclosures of nonlinear mapping with degenerate ellipsoids. February 2024.

[62] Morgan Louédec, Luc Jaulin, and Christophe Viel. Computational tractable guaranteed numerical method to study the stability of n-dimensional time-independent nonlinear systems with bounded perturbation. *Automatica*, 153:110981, July 2023.

[63] Morgan Louédec, Luc Jaulin, and Christophe Viel. A guaranteed numerical method to prove the exponential stability of nonlinear discrete-time systems. February 2024.

[64] A. M. LYAPUNOV. *The general problem of the stability of motion.* PhD thesis, Univ. Kharkov, 1892.

[65] V M Marchenko. Hybrid Discrete-Continuous Control Systems: I. Representation of Solutions. 50(11), 2014.

[66] Gian Luca Mariottini, Fabio Morbidi, Domenico Prattichizzo, Nicholas Vander Valk, Nathan Michael, George Pappas, and Kostas Daniilidis. Vision-Based Localization for Leader–Follower Formation Control. *IEEE Transactions on Robotics*, 25(6):1431–1438, December 2009. Conference Name: IEEE Transactions on Robotics.

[67] Francesca Mazzia and Alessandra Sestini. On a class of Hermite-Obreshkov one-step methods with continuous spline extension. *axioms*, 7(3):58, 2018. ISBN: 2075-1680 Publisher: MDPI.

[68] Farhad Mehdifar, Charalampos P. Bechlioulis, Farzad Hashemzadeh, and Mahdi Baradarannia. Prescribed performance distance-based formation control of Multi-Agent Systems. *Automatica*, 119:109086, September 2020.

[69] Robert E. Melchers. The effect of corrosion on the structural reliability of steel offshore structures. *Corrosion Science*, 47(10):2391–2410, October 2005.

[70] Florian Messerer and Moritz Diehl. An Efficient Algorithm for Tube-based Robust Nonlinear Optimal Control with Optimal Linear Feedback. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 6714–6721, December 2021. ISSN: 2576-2370.

[71] Konstantin Mischaikow and Marian Mrozek. Chaos in the Lorenz equations: a computer-assisted proof. *Bulletin of the American Mathematical Society*, 32(1):66–72, 1995.

[72] Ramon E. Moore. The automatic analysis and control of error in digital computation based on the use of interval numbers. *Error in digital computation*, 1:61–130, 1965.

[73] Ramon E. Moore. *Methods and applications of interval analysis.* SIAM, 1979.

[74] Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud. *Introduction to Interval Analysis.* Society for Industrial and Applied Mathematics, January 2009.

[75] Fabio Morbidi, Gian Luca Mariottini, and Domenico Prattichizzo. Observer design via Immersion and Invariance for vision-based leader–follower formation control. *Automatica*, 46(1):148–154, January 2010.

[76] Markus Mueller and Robin Wallace. Enabling science and technology for marine renewable energy. *Energy Policy*, 36(12):4376–4382, December 2008.

[77] Robin R. Murphy. Designing a robot to recover a sunken submarine is hard. *Science Robotics*, 8(81), August 2023. Publisher: American Association for the Advancement of Science.

[78] Nedialko S. Nedialkov and Kenneth R. Jackson. An interval Hermite-Obreschkoff method for computing rigorous bounds on the solution of an initial value problem for an ordinary differential equation. *Reliable Computing*, 5(3):289–310, 1999. ISBN: 1385-3139 Publisher: Springer.

[79] Dragan Nesi and Andrew R Teel. Sampled-data control of nonlinear systems: An overview of recent results. 2007.

[80] Nikolaos Nikolakis, Vasilis Maratos, and Sotiris Makris. A cyber physical system (CPS) approach for safe human-robot collaboration in a shared workplace. *Robotics and Computer-Integrated Manufacturing*, 56:233–243, April 2019.

[81] Cf ODDS. Transforming our world: the 2030 Agenda for Sustainable Development. Technical report, United Nations, New York, NY, USA, 2015.

[82] Kwang-Kyo Oh and Hyo-Sung Ahn. Distance-based undirected formations of single-integrator and double-integrator modeled agents in n-dimensional space. *International Journal of Robust and Nonlinear Control*, 24(12):1809–1820, 2014.

[83] Kwang-Kyo Oh, Myoung-Chul Park, and Hyo-Sung Ahn. A survey of multi-agent formation control. *Automatica*, 53:424–440, March 2015.

[84] Reza Olfati-Saber, J. Alex Fax, and Richard M. Murray. Consensus and Cooperation in Networked Multi-Agent Systems. *Proceedings of the IEEE*, 95(1):215–233, January 2007. Conference Name: Proceedings of the IEEE.

[85] Min Gyu Park, Jae Hyun Jeon, and Min Cheol Lee. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing. In *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No.01TH8570)*, volume 3, pages 1530–1535 vol.3, June 2001.

[86] Boris T. Polyak, Alexander V. Nazin, Michael V. Topunov, and Sergey A. Nazin. Rejection of Bounded Disturbances via Invariant Ellipsoids Technique. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 1429–1434, December 2006. ISSN: 0191-2216.

[87] Stephen B Pope. Algorithms for Ellipsoids. page 49, 2008.

[88] James Preisig. Acoustic propagation considerations for underwater acoustic communications network development. *ACM SIGMOBILE Mobile Computing and Communications Review*, 11(4):2–10, October 2007.

[89] Nacim Ramdani, Nacim Meslem, and Yves Candau. A Hybrid Bounding Method for Computing an Over-Approximation for the Reachable Set of Uncertain Nonlinear Systems. *IEEE Transactions on Automatic Control*, 54(10):2352–2364, October 2009. Conference Name: IEEE Transactions on Automatic Control.

[90] Nacim Ramdani and Nedialko S. Nedialkov. Computing reachable sets for uncertain nonlinear hybrid systems using interval constraint propagation techniques. *IFAC Proceedings Volumes*, 42(17):156–161, January 2009.

[91] Nacim Ramdani, Louise Travé-Massuyès, and Carine Jauberthie. Mode discernibility and bounded-error state estimation for nonlinear hybrid systems. *Automatica*, 91:118–125, May 2018.

[92] Andreas Rauh, Auguste Bourgois, and Luc Jaulin. Verifying Provable Stability Domains for Discrete-Time Systems Using Ellipsoidal State Enclosures. *Acta Cybernetica*, May 2022. Publisher: University of Szeged.

[93] Andreas Rauh, Auguste Bourgois, and Luc Jaulin. Verifying Provable Stability Domains for Discrete-Time Systems Using Ellipsoidal State Enclosures. *Acta Cybernetica*, May 2022.

[94] Andreas Rauh, Eberhard P. Hofer, and Ekaterina Auer. ValEncIA-IVP: A comparison with other initial value problem solvers. In *12th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN 2006)*, pages 36–36. IEEE, 2006.

[95] Andreas Rauh and Luc Jaulin. A computationally inexpensive algorithm for determining outer and inner enclosures of nonlinear mappings of ellipsoidal domains. *International Journal of Applied Mathematics and Computer Science*, 31(3):399–415, 2021.

[96] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, pages 25–34, New York, NY, USA, August 1987. Association for Computing Machinery.

[97] Simon Rohou and Benoît Desrochers. The Codac library – Constraint-programming for robotics, 2022.

[98] Swantje Romig, Luc Jaulin, and Andreas Rauh. Using Interval Analysis to Compute the Invariant Set of a Nonlinear Closed-Loop Control System. *Algorithms*, 12(12):262, December 2019. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[99] Yuki Sekimori, Yang Weng, Takumi Matsuda, Yukiyasu Noguchi, Chihaya Kawamura, and Toshihiro Maki. Acoustic Passive BEDD Self-localization for A Fleet of AUVs: A Sea Experiment Validation. In *OCEANS 2022, Hampton Roads*, pages 1–8, October 2022. ISSN: 0197-7385.

[100] Jianhua Shi, Jiafu Wan, Hehua Yan, and Hui Suo. A survey of Cyber-Physical Systems. In *2011 International Conference on Wireless Communications and Signal Processing (WCSP)*, pages 1–6, November 2011.

[101] Walter H. F. Smith and Karen M. Marks. Seafloor in the Malaysia Airlines Flight MH370 Search Area. *Eos, Transactions American Geophysical Union*, 95(21):173–174, May 2014.

[102] Sergey Staroletov. Automatic Proving of Stability of the Cyber-Physical Systems in the Sense of Lyapunov with KeYmaera. 2021.

[103] Tor Harald Staurnes, Pål Kristoffer Kjærem, Lauritz Rismark Fosso, and Kristian Aasmundstad Johannesen. *Decentralized Model Predictive Control for Increased Autonomy in Fleets of ROVs.* Bachelor thesis, NTNU, 2023. Accepted: 2023-07-12T17:21:26Z.

[104] Zhiyong Sun, Héctor Garcia de Marina, Georg S. Seyboth, Brian D. O. Anderson, and Changbin Yu. Circular Formation Control of Multiple Unicycle-Type Agents With Nonidentical Constant Speeds. *IEEE Transactions on Control Systems Technology*, 27(1):192–205, January 2019. Conference Name: IEEE Transactions on Control Systems Technology.

[105] Auwal Shehu Tijjani, Ahmed Chemori, and Vincent Creuze. A survey on tracking control of unmanned underwater vehicles: Experiments-based approach. *Annual Reviews in Control*, September 2022.

[106] Norman Toro, Pedro Robles, and Ricardo I. Jeldres. Seabed mineral resources, an alternative for the future of renewable energy: A critical review. *Ore Geology Reviews*, 126:103699, November 2020.

[107] Warwick Tucker. The Lorenz attractor exists. *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, 328(12):1197–1202, June 1999.

[108] Petra Valiauga, Xuhui Feng, Mario E. Villanueva, Radoslav Paulen, and Boris Houska. Set-membership Estimation using Ellipsoidal Ensembles. *IFAC-PapersOnLine*, 54(3):596–601, January 2021.

[109] Giorgio Valmorbida and James Anderson. Region of attraction estimation using invariant sets and rational Lyapunov functions. *Automatica*, 75:37–45, January 2017.

[110] C. X. S. Viel. *Control law and state estimators design for multi-agent system with reduction of communications by event-triggered approach.* Thesis, January 2018. Accepted: 2018-01-11T09:42:04Z.

[111] Christophe Viel, Michel Kieffer, Hélène Piet-Lahanier, and Sylvain Bertrand. Distributed event-triggered formation control for multi-agent systems in presence of packet losses. *Automatica*, 141:110215, July 2022.

[112] Irmina Walawska and Daniel Wilczak. An implicit algorithm for validated enclosures of the solutions to variational equations for ODEs. *Applied Mathematics and Computation*, 291:303–322, December 2016.

[113] Linling Wang, Daqi Zhu, Wen Pang, and Youmin Zhang. A survey of underwater search for multi-target using Multi-AUV: Task allocation, path planning, and formation control. *Ocean Engineering*, 278:114393, June 2023.

[114] Xiaomin Wang, Benoit Zerr, Héléne Thomas, Benoit Clement, and Zexiao Xie. Pattern formation of multi-AUV systems with the optical sensor based on displacement-based formation control. *International Journal of Systems Science*, 51(2):348–367, January 2020.

[115] Daniel Wilczak and Piotr Zgliczyński. C^{r}-Lohner algorithm. *Schedae Informaticae*, 20, 2011. ISBN: 1732-3916.

[116] Luo Xu, Zhongguan Wang, and Hongbin Sun. Modeling of Time-Delayed Distributed Cyber-Physical Power Systems for Small-Signal Stability Analysis. *IEEE Transatctions on Smart Grid*, 12(4), 2021.

[117] Jing Yan, Jin Gao, Xian Yang, Xiaoyuan Luo, and Xinping Guan. Position Tracking Control of Remotely Operated Underwater Vehicles With Communication Delay. *IEEE Transactions on Control Systems Technology*, 28(6):2506–2514, November 2020. Conference Name: IEEE Transactions on Control Systems Technology.

[118] Tao Yan, Zhe Xu, Simon X Yang, and S Andrew Gadsden. Formation control of multiple autonomous underwater vehicles: a review. 2023.

[119] Yue Yang, Yang Xiao, and Tieshan Li. A Survey of Autonomous Underwater Vehicle Formation: Performance, Formation Control, and Communication Capability. *IEEE Communications Surveys & Tutorials*, 23(2):815–841, 2021. Conference Name: IEEE Communications Surveys & Tutorials.

[120] Chika Yoshioka and Toru Namerikawa. Formation Control of Nonholonomic Multi-Vehicle Systems based on Virtual Structure. *IFAC Proceedings Volumes*, 41(2):5149–5154, January 2008.

[121] Dongmin Yu, Yingying Mao, Bing Gu, Sayyad Nojavan, Kittisak Jermsittiparsert, and Maryam Nasseri. A new LQG optimal control strategy applied on a hybrid wind turbine/solid oxide fuel cell/ in the presence of the interval uncertainties. *Sustainable Energy, Grids and Networks*, 21:100296, March 2020.

[122] Piotr Zgliczynski. C1Lohner Algorithm. *Foundations of Computational Mathematics*, 2(4):429–465, October 2002.

[123] Chenyu Zhao, Philipp R. Thies, and Lars Johanning. Offshore inspection mission modelling for an ASV/ROV system. *Ocean Engineering*, 259:111899, September 2022.

[124] Yuanshi Zheng, Qi Zhao, Jingying Ma, and Long Wang. Second-order consensus of hybrid multi-agent systems. *Systems & Control Letters*, 125:51–58, March 2019.

Appendix

Computation of $\boldsymbol{h}\left(\boldsymbol{m}_k\right)$

This section presents the computation of $\boldsymbol{h}\left(\boldsymbol{m}_k\right)$ from Section 5.6.2.1. From (5.60), one has

$$\boldsymbol{m}_{k+1} = \begin{bmatrix} d_{b,k+1} - d^* \\ d_{r,k+1} - d^* \\ \phi_{r,k+1} - \phi_{b,k+1} - \frac{3}{\pi} \\ v_{b,k+1} \\ v_{r,k+1} \\ w_{b,k+1} \\ w_{r,k+1} \end{bmatrix}. \tag{10.1}$$

Thus, with (5.55), one deduces

$$\boldsymbol{m}_{k+1} = \begin{bmatrix} d_{b,k} - d^* + T \cdot v_{b,k} \\ d_{r,k} - d^* + T \cdot v_{r,k} \\ (\phi_{r,k} + T \cdot w_{r,k}) - (\phi_{b,k} + T \cdot w_{b,k}) - \frac{3}{\pi} \\ v_{b,k} + T \cdot u_{1,k} \\ v_{r,k} + T \cdot u_{2,k} \\ w_{b,k} + T \cdot \frac{u_{3,k}}{d_{b,k}} \\ w_{r,k} + T \cdot \frac{u_{4,k}}{d_{r,k}} \end{bmatrix}. \tag{10.2}$$

Moreover, with (5.59), one gets

$$\boldsymbol{m}_{k+1} = \begin{bmatrix} d_{b,k} - d^* + T \cdot v_{b,k} \\ d_{r,k} - d^* + T \cdot v_{r,k} \\ \left(\phi_{r,k} - \phi_{b,k} - \frac{3}{\pi}\right) + T \cdot (w_{r,k} - w_{b,k}) \\ v_{b,k} + T \cdot s \cdot \arctan\left(k_{p,d} \cdot (d^* - d_{b,k}) - k_{d,d} \cdot v_{b,k}\right) \\ v_{r,k} + T \cdot s \cdot \arctan\left(k_{p,d} \cdot (d^* - d_{r,k}) - k_{d,d} \cdot v_{r,k}\right) \\ w_{b,k} + \frac{T \cdot s}{d_{b,k}} \arctan\left(k_{p,\phi} \cdot \left(\phi_k - \frac{\pi}{6} - \phi_{b,k}\right) - k_{d,\phi} \cdot w_{b,k}\right) \\ w_{r,k} + \frac{T \cdot s}{d_{r,k}} \arctan\left(k_{p,\phi} \cdot \left(\phi_k + \frac{\pi}{6} - \phi_{r,k}\right) - k_{d,\phi} \cdot w_{r,k}\right) \end{bmatrix}. \tag{10.3}$$

Then, from (5.58), one also has

$$\begin{aligned}\phi_k - \frac{\pi}{6} - \phi_{b,k} &= \frac{\phi_{b,k} + \phi_{r,k}}{2} - \frac{\pi}{6} - \phi_{b,k}, \\ &= \frac{1}{2}\left(\phi_{r,k} - \phi_{b,k} - \frac{\pi}{3}\right), \\ &= \frac{1}{2} m_{3,k}, \end{aligned} \tag{10.4}$$

and

$$\begin{aligned}\phi_k + \frac{\pi}{6} - \phi_{r,k} &= \frac{\phi_{b,k} + \phi_{r,k}}{2} + \frac{\pi}{6} - \phi_{r,k}, \\ &= -\frac{1}{2}\left(\phi_{r,k} - \phi_{b,k} - \frac{\pi}{3}\right), \\ &= \frac{1}{2} m_{3,k}. \end{aligned} \tag{10.5}$$

Finlay, from (5.60), (10.3), (10.4) and (10.5), one obtains

$$\begin{aligned}\boldsymbol{h}\left(\boldsymbol{m}_k\right) &= \boldsymbol{m}_{k+1}, \\ &= \begin{bmatrix} m_{1,k} + T \cdot m_{4,k} \\ m_{2,k} + T \cdot m_{5,k} \\ m_{3,k} + T \cdot \left(m_{7,k} - m_{6,k}\right) \\ m_{4,k} + sT \cdot \arctan\left(-k_{p,d} \cdot m_{1,k} - k_{d,d} \cdot m_{4,k}\right) \\ m_{5,k} + sT \cdot \arctan\left(-k_{p,d} \cdot m_{2,k} - k_{d,d} \cdot m_{5,k}\right) \\ m_{6,k} + \frac{sT}{m_{1,k}+d^*} \arctan\left(\frac{k_{p,\phi}}{2} m_{3,k} - k_{d,\phi} \cdot m_{6,k}\right) \\ m_{7,k} + \frac{sT}{m_{2,k}+d^*} \arctan\left(-\frac{k_{p,\phi}}{2} m_{3,k} - k_{d,\phi} \cdot m_{7,k}\right) \end{bmatrix}. \end{aligned} \tag{10.6}$$

Elements of the Jacobian matrix

This section details the elements of the Jacobian matrix from (5.63). One has

$$J_{4,1} = \frac{-sTk_{p,d}}{\left(-k_{p,d} \cdot m_1 - k_{d,d} \cdot m_4\right)^2 + 1} \tag{10.7}$$

$$J_{4,4} = 1 - \frac{sTk_{d,d}}{\left(-k_{p,d} \cdot m_1 - k_{d,d} \cdot m_4\right)^2 + 1} \tag{10.8}$$

$$J_{5,2} = \frac{-sTk_{p,d}}{\left(-k_{p,d} \cdot m_2 - k_{d,d} \cdot m_5\right)^2 + 1} \tag{10.9}$$

$$J_{5,5} = 1 - \frac{sTk_{d,d}}{\left(-k_{p,d} \cdot m_2 - k_{d,d} \cdot m_5\right)^2 + 1} \tag{10.10}$$

$$J_{6,1} = \frac{-sT}{\left(m_1 + d^*\right)^2} \arctan\left(\frac{k_{p,\phi}}{2} m_3 - k_{d,\phi} \cdot m_6\right) \tag{10.11}$$

$$J_{6,3} = \frac{sTk_{p,\phi}}{2\left(m_1 + d^*\right)\left(\left(\frac{k_{p,\phi}}{2} m_3 - k_{d,\phi} \cdot m_6\right)^2 + 1\right)} \tag{10.12}$$

$$J_{6,6} = 1 - \frac{sTk_{d,\phi}}{\left(m_1 + d^*\right)\left(\left(\frac{k_{p,\phi}}{2} m_3 - k_{d,\phi} \cdot m_6\right)^2 + 1\right)} \tag{10.13}$$

$$J_{7,2} = \frac{-sT}{\left(m_2 + d^*\right)^2} \arctan\left(-\frac{k_{p,\phi}}{2} m_3 - k_{d,\phi} \cdot m_7\right) \tag{10.14}$$

$$J_{7,3} = \frac{-sTk_{p,\phi}}{2\left(m_2 + d^*\right)\left(\left(-\frac{k_{p,\phi}}{2} m_3 - k_{d,\phi} \cdot m_7\right)^2 + 1\right)} \tag{10.15}$$

$$J_{7,7} = 1 - \frac{sTk_{d,\phi}}{\left(m_2 + d^*\right)\left(\left(-\frac{k_{p,\phi}}{2} m_3 - k_{d,\phi} \cdot m_7\right)^2 + 1\right)} \tag{10.16}$$

Proof of Equation (7.39)

From Theorem 7.2, consider the ellipsoid $\mathscr{E}$, the symmetric matrix $\boldsymbol{M}$, the orthonormal matrix $\boldsymbol{U}$, the diagonal matrix $\boldsymbol{S} = \operatorname{diag}\left(s_i\right)_{i \in [\![1,n]\!]}$, the point $\boldsymbol{\mu}_{\text{out}}$ and $\boldsymbol{x} \in \mathscr{E}$, and the unitary vector of the Cartesian base $\boldsymbol{e}_i$. Let us write

$$\begin{aligned} \boldsymbol{c} &= \boldsymbol{U}^T \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right), \\ &= \left(c_i\right)_{i \in [\![1,n]\!]}. \end{aligned} \tag{10.17}$$

From (7.32), One has

$$\begin{aligned} \boldsymbol{M}\boldsymbol{M}^+ \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right) &= \left(\boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^T\right)\left(\boldsymbol{U}\boldsymbol{S}^+\boldsymbol{U}^T\right)\left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right), \\ &= \boldsymbol{U}\boldsymbol{S}\boldsymbol{S}^+\boldsymbol{U}^T \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}}\right), \\ &= \boldsymbol{U}\boldsymbol{S}\boldsymbol{S}^+\boldsymbol{c}. \end{aligned} \tag{10.18}$$

Moreover, from (7.34), $\boldsymbol{S}\boldsymbol{S}^+$ is diagonal and its elements are

$$s_i s_i^+ = \begin{cases} 0 & \text{if} \max_{\boldsymbol{x} \in \mathscr{E}} \left| \boldsymbol{e}_i^T \boldsymbol{U}^T \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}} \right) \right| = 0 \\ 1 & \text{else} \end{cases}$$

Thus, if $s_i s_i^+ = 0$, then $c_i = 0$. As a result, for all $i \in [\![1, n]\!]$, one obtains

$$s_i s_i^+ c_i = c_i.$$

So, one gets $\boldsymbol{S}\boldsymbol{S}^+\boldsymbol{c} = \boldsymbol{c}$. Thus, one deduces

$$\begin{aligned} \boldsymbol{M}\boldsymbol{M}^+ \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}} \right) &= \boldsymbol{U}\boldsymbol{S}\boldsymbol{S}^+\boldsymbol{c}, \\ &= \boldsymbol{U}\boldsymbol{c}, \\ &= \boldsymbol{U}\boldsymbol{U}^T \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}} \right), \\ &= \left(\boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\mu}_{\text{out}} \right). \end{aligned} \tag{10.19}$$

Computation of $\boldsymbol{f}_x$

From (8.46), one has

$$\begin{aligned} u_{3,j} &= s \cdot \arctan\left(k_{p,\phi} \cdot \left(\phi_j - \frac{\pi}{6} - \phi_{b,j} \right) \right) && (10.20) \\ &\overset{(8.45)}{=} s \cdot \arctan\left(k_{p,\phi} \cdot \left(\frac{\phi_{b,j} + \phi_{r,j}}{2} - \frac{\pi}{6} - \phi_{b,j} \right) \right) && (10.21) \\ &= && (10.22) \\ &\overset{(8.48)}{=} s \cdot \arctan\left(k_{p,\phi} \cdot \frac{m_{3,j}}{2} \right) && (10.23) \end{aligned}$$

Similarly, one has

$$\begin{aligned} u_{4,j} &= -u_{3,j}. \\ &= s \cdot \arctan\left(k_{p,\phi} \cdot \frac{-m_{3,j}}{2} \right) \end{aligned} \tag{10.24}$$

Moreover, one has

$$\begin{aligned}
\boldsymbol{f}_x\left(\boldsymbol{x},\boldsymbol{m}_j\right) &= \dot{\boldsymbol{x}} \\
&\overset{(8.47)}{=} \begin{bmatrix} \dot{d}_b \\ \dot{d}_r \\ \left(\dot{\phi}_r - \dot{\phi}_b\right) \\ \ddot{d}_b \\ \ddot{d}_r \\ \ddot{\phi}_b \\ \ddot{\phi}_r \end{bmatrix} \\
&\overset{(8.42)}{=} \begin{bmatrix} \dot{d}_b \\ \dot{d}_r \\ \left(\dot{\phi}_r - \dot{\phi}_b\right) \\ u_{1,j} - c\cdot\dot{d}_b \\ u_{2,j} - c\cdot\dot{d}_r \\ \frac{u_{3,j}}{d_b} - c\cdot\dot{\phi}_b \\ \frac{u_{4,j}}{d_r} - c\cdot\dot{\phi}_r \end{bmatrix} \\
&\overset{(8.46),(10.23),(10.24)}{=} \begin{bmatrix} \dot{d}_b \\ \dot{d}_r \\ \left(\dot{\phi}_r - \dot{\phi}_b\right) \\ s\cdot\arctan\left(k_{p,d}\cdot\left(d^* - d_{b,j}\right)\right) - c\cdot\dot{d}_b \\ s\cdot\arctan\left(k_{p,d}\cdot\left(d^* - d_{r,j}\right)\right) - c\cdot\dot{d}_r \\ \frac{s\cdot\arctan\left(k_{p,\phi}\cdot\frac{m_{3,j}}{2}\right)}{d_b} - c\cdot\dot{\phi}_b \\ \frac{-s\cdot\arctan\left(k_{p,\phi}\cdot\frac{m_{3,j}}{2}\right)}{d_r} - c\cdot\dot{\phi}_r \end{bmatrix} \\
&\overset{(8.47),(8.48)}{=} \begin{bmatrix} x_4 \\ x_5 \\ \left(x_7 - x_6\right) \\ -s\cdot\arctan\left(k_{p,d}\cdot m_{1,j}\right) - c\cdot x_4 \\ -s\cdot\arctan\left(k_{p,d}\cdot m_{2,j}\right) - c\cdot x_5 \\ \frac{1}{x_1+d^*}s\cdot\arctan\left(k_{p,\phi}\cdot\frac{m_{3,j}}{2}\right) - c\cdot x_6 \\ \frac{-1}{x_2+d^*}s\cdot\arctan\left(k_{p,\phi}\cdot\frac{m_{3,j}}{2}\right) - c\cdot x_7 \end{bmatrix}.
\end{aligned} \tag{10.25}$$

Computation of $\boldsymbol{h}_{m,q_b}$ and $\boldsymbol{h}_{m,q_r}$

One has

$$\begin{aligned}
\boldsymbol{h}_{m,q_b}\left(\boldsymbol{x}\left(t_{j+1}\right),\boldsymbol{m}_j,\boldsymbol{v}\left(t_{j+1}\right)\right) &= \boldsymbol{m}_{j+1} \\
&= \begin{bmatrix} d_{b,j+1} - d^*, \\ d_{r,j+1} - d^*, \\ \phi_{r,j+1} - \phi_{b,j+1} - \frac{\pi}{3}. \end{bmatrix}.
\end{aligned} \tag{10.26}$$

This mapping is applied when j is odd. So from (8.41), one has

$$\begin{aligned} d_{b,j+1} &= d_b\left(t_{j+1}\right), \\ \phi_{b,j+1} &= \phi_b\left(t_{j+1}\right), \\ d_{r,i+1} &= d_{r,j}, \\ \phi_{r,i+1} &= \phi_{r,j}. \end{aligned} \tag{10.27}$$

As a result,

$$\begin{aligned} \boldsymbol{h}_{m,q_b}\left(\boldsymbol{x}\left(t_{j+1}\right), \boldsymbol{m}_j, \boldsymbol{v}\left(t_{j+1}\right)\right) &= \left[\begin{array}{c} d_b\left(t_{j+1}\right) - d^*, \\ d_{r,j} - d^*, \\ \phi_{r,j} - \phi_b\left(t_{j+1}\right) - \frac{\pi}{3}. \end{array}\right] \\ &= \left[\begin{array}{c} x_1\left(t_{j+1}\right) \\ m_{2,j} \\ x_3\left(t_{j+1}\right) + \phi_{r,j} - \phi_r\left(t_{j+1}\right) \end{array}\right] \\ &= \left[\begin{array}{c} x_1\left(t_{j+1}\right) \\ m_{2,j} \\ x_3\left(t_{j+1}\right) - v_2\left(t_{j+1}\right) \end{array}\right]. \end{aligned} \tag{10.28}$$

The mapping $\boldsymbol{h}_{m,q_r}$ is obtained with a similar reasoning.

Computation of $\boldsymbol{f}_v$, $\boldsymbol{h}_{v,q_b}$ and $\boldsymbol{h}_{v,q_r}$

Since

$$\boldsymbol{v}(t) = \left[\begin{array}{c} \phi_b(t) - \phi_{b,j} \\ \phi_r(t) - \phi_{r,j} \end{array}\right], \ \forall t \in \left[t_j^+, t_{j+1}\right], \tag{10.29}$$

then

$$\begin{aligned} \boldsymbol{f}_v\left(\boldsymbol{x}(t)\right) &= \dot{\boldsymbol{v}}(t) \\ &= \left[\begin{array}{c} \dot{\phi}_b(t) \\ \dot{\phi}_r(t) \end{array}\right], \\ &= \left[\begin{array}{c} x_6(t) \\ x_7(t) \end{array}\right]. \end{aligned} \tag{10.30}$$

Moreover, for every odd $j \in \mathbb{N}$ one has

$$\begin{aligned} \boldsymbol{h}_{v,q_b}\left(\boldsymbol{v}\left(t_j\right)\right) &= \boldsymbol{v}\left(t_j^+\right) \\ &= \left[\begin{array}{c} \phi_b(t) - \phi_{b,j} \\ \phi_r(t) - \phi_{r,j} \end{array}\right] \end{aligned} \tag{10.31}$$

Since j is odd, one has

$$\begin{aligned} \phi_{b,j} &= \phi_b(t), \\ \phi_{r,j} &= \phi_{r,j-1}. \end{aligned}$$

Therefore one gets

$$\begin{aligned}\boldsymbol{h}_{v,q_b}\left(\boldsymbol{v}\left(t_j\right)\right) &= \left[\begin{array}{c} 0 \\ \phi_r\left(t\right) - \phi_{r,j-1} \end{array}\right] \\ &= \left[\begin{array}{c} 0 \\ v_2\left(t_j\right) \end{array}\right].\end{aligned}$$

The same reasoning can be applied on $\boldsymbol{h}_{v,q_r}$.