SAPIENZA
UNIVERSITÀ DI ROMA

Formulation and applications of a variational feedback control technique

Università di Roma, Sapienza

Dottorato di Ricerca in Applied and theoretical Mechanical Engineering – XXXII Ciclo

Candidate

Leandro Nesi
ID number 1318700

Thesis Advisor

Prof. Antonio Carcaterra

Co-Advisor

Dr. Gianluca Pepe

September 2019

Formulation and applications of a variational feedback control technique
Ph.D. thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Version: 14-Feb-2020

Author's email: leandro.nesi@uniroma1.it

Dedicated to
Deborah, the love of my life.
"When the sun has set, no candle can replace it".

Abstract

This work originates from the desire to produce innovative control for engineering applications in the field of Autonomous Vehicles (AVs).
Nowadays, many controllers are present in the literature related to the AVs, from classical ones such as the Linear Quadratic Regulator to Model Predictive Controls techniques and Direct and Indirect methods based on Optimal Control Theories.
Today used theories present some problems from an engineering point of view since they can be very demanding regarding the computational costs, or they may have some strict limitations in the mathematical form of the model.
In this work, the goal was to find a reliable, robust and flexible control, which may be used with different dynamical systems providing an excellent fail-safe strategy. The flexibility requirement is due the versatility of Autonomous Vehicles regarding both models and applications.
A new method for controlling nonlinear dynamical systems is here introduced, named Feedback Local Optimality Principle – FLOP. The FLOP control is based on Optimal Control Theory and it tries to overcome some of the main difficulties of the classical theory and of its numerical method of solutions, i.e. directs and indirect methods.
The FLOP control succeeds in providing a feedback control law for nonlinear dynamical systems, and its possible applications go from the control of ground vehicles, such as unicycles and autonomous cars to aerial and marine unmanned vehicles. In this thesis, the applicability of the FLOP control to Swarm Robotics applications is shown. In particular, the chance to control a high number of nonlinear agents with different individual and collective tasks is explained deeply.
The FLOP control, by the acceptance of a local optimum to a variational problem, is able to provide a feedback control law for a wide class of possible applications. It is a control that is easy to test since it does not require huge tuning-time, it has low computational costs and therefore it can be also used in real-time applications. In this work it is tested on numerical examples and simulations, but further developments will show its applicability also in experimental situations.

Ringraziamenti

Ho deciso di scrivere i ringraziamenti in italiano, nella speranza di riuscire a dire il grazie più giusto possibile alle persone che hanno fatto sì io potessi arrivare a chiudere questo percorso. Prima di tutto voglio ringraziare il mio relatore, il professor Antonio Carcaterra. Ha saputo guidarmi, consigliarmi e soprattutto ha voluto capirmi in questi tre anni di dottorato. Mi ha spronato, stimolato e fatto sentire a casa per tanto tempo. Ha avuto fiducia in me quando non pensavo ne avrei mai avuta in università. Gli sarò sempre grato.
Voglio ringraziare il mio correlatore, l'ingegner Gianluca Pepe. Non è stato un percorso facile e sicuramente non è stato lineare: ci siamo anche scontrati, ma esco da questi tre anni rafforzato, convinto di esser cresciuto e migliorato sotto la sua guida, come ingegnere e come persona.
Un grazie va ai miei compagni di avventura. Dario, con cui è nata un'amicizia che sono sicuro troverà il modo di proseguire. Federica, un'amica schietta e sincera che non ha mai fatto mancare il suo parere, anche brutale ma sempre onesto, quando richiesto; quella stessa persona di cui ho una stima professionale indicibile e che non vedo l'ora veder spaccare il mondo in due. Elena, molto più che una collega. Amir, la mia compagnia preferita al pub per distacco. Manuel, con cui abbiamo iniziato ormai tanto tempo fa ma siamo andati sempre più in profondità. Maicol, una vera e propria sorpresa, che ha saputo rendere il mio periodo lavorativamente più intenso molto meno duro. A lui dico di continuare a tirar fuori la meravigliosa persona che è. Ilaria, per l'allegria che porta in stanza. Giocerra, per essersi fidata al buio di me ed essersi tramutata in un'amicizia vera, quella di cui non si può fare a meno.
Grazie ai miei genitori. Papà, da sempre un punto di riferimento e un modello da seguire, sempre pronto a venire in soccorso quando veramente contava, sempre sincero. Fra padri e figli non dovrebbero esistere segreti, e da qualche tempo a questa parte stiamo imparando a non averne. Troveremo sempre un punto di contatto.
Mamma, che c'è sempre. C'è sempre in un modo che neanche si riesce ad immaginare, che poi è la potenza delle mamme. Nel momento di massimo bisogno, era lì che parlava a cuore aperto con Deborah. Per sempre grato.
Grazie a Mimmo e al suo modo di esserci, di spronare, di lottare perchè io trovi me stesso, perchè io creda in me, perchè io sia la versione migliore possibile di me stesso. La strada davanti è molto più lunga della lunghissima strada già percorsa.
Grazie a Giovanni, la mia ispirazione più grande. Lavoratore mostruoso, talento fenomenale, carattere formidabile. Il tuo inseguire i tuoi sogni è quel che ogni giorno mi sprona ad inseguire i miei. Ci sarà sempre un vino da stappare, un sous vide da fare, una langa da esplorare.
Agli amici che oramai sono gli amici della vita:
Matteo, con cui ho condiviso i segreti e le paure più grandi. A Ludovica, senza la quale i segreti e le paure forse non ce le saremmo riusciti a raccontare.
Dom, che mi mostra la via.
Luca, con il quale non colmerò mai il mio debito più grande.
Vale, l'amica di cui avrò sempre bisogno.
Vincenzo e Antonino, la definizione dell'amore che vince sempre e dell'amicizia senza età.

Alessio, cui auguro di essere felice nel privato come lo sono io.
Domenico e Michela, due esseri umani straordinari.
Stefano, per avermi preso per mano e mostrato un mondo che ignoravo.
Cristina, per ricordarmi che il talento, quando lavora duro, non vede limiti.
A Raffaella e Lavinia, due tifose vere, appassionate, cui voglio un bene profondissimo.
A Federico, con cui abbiamo condiviso il meglio e il peggio l'uno dell'altro ma siamo ancora qui a chiamarci e sederci davanti ad una birra.
Alla mia seconda famiglia, Claudio e Lilly. Secondi genitori per davvero, per farmi sentire amato nel quotidiano.
Ai miei nonni, che rimangono uno dei più grandi privilegi che potrò mai dire di aver avuto.
A zio Marco e zia Cecilia, per avermi ricordato che la Famiglia è un valore aggiunto indescrivibile.
A Vellut e Divi, una sorella e un fratello acquisiti. So che con Stefano siamo solo agli inizi, ma è un grandissimo inizio. Vi auguro il meglio del meglio del meglio.
A Paolo, che mi ha insegnato a chi dare importanza.
A Marco, senza il quale la mia vita sarebbe stata diversa e senza il quale io oggi non sarei qui. L'impatto che hai avuto nella mia vita è stato straordinario. In ogni cosa che faccio, per ogni traguardo che ho raggiunto, per ogni soddisfazione che mi sono levato, non ho mai dimenticato che tu mi abbia, senza mezzi termini, regalato tantissimo della vita che ho vissuto. Non ti avessi incontrato non sarei qui a festeggiare. Grazie dal più profondo di me.
Ai ragazzi di Overtime: mi state regalando un sogno che non pensavo avrei mai potuto vivere. Ho trovato persone speciali, tutte diverse, ognuna mi ha insegnato qualcosa e stiamo solo grattando la superficie.
Luca, hai creduto in me senza avermi mai neanche conosciuto. Da quel giorno ho capito che la passione per qualcosa può trasformare il proprio mondo, e non ho mai smesso di far trasformare il mio. Non ho mai smesso di mettere passione e impegno nelle cose che mi fanno venir voglia di non dormire la notte, ed è forse uno dei risultati migliori che ho avuto.
Simo Rancid, mi hai insegnato a non accontentarmi della superficie. C'è sempre qualcosa in più da conoscere. C'è sempre qualcosa da capire meglio, si può sempre dar di più. E si possono sempre guardare le cose da tante angolazioni, ma ne esiste sempre una migliore dell'altra.
Simo Severi. Un fratello. Un biglietto di treno preso, scoprirsi più simili di quel che le nostre diversità lasciavano pensare. Mi hai insegnato la cosa più importante di tutte: 'less is more'. Mi ricordi di respirare e di vivere a pieni polmoni e a tutta vita.
A Tony. Te l'ho promesso in una notte canadese, in cui vedevo ancora più ombre che luci: "Never stop shooting for the stars".
A Deborah.
Sei esattamente quel che ho sempre pensato saresti diventata e non potrei mai chiedere di meglio. Con tutti i nostri difetti, con tutta la strada che abbiamo da compiere, sei perfetta per me. Abbiamo visto posti, da Budapest a Bruges, da Dublino a Washington, da Parigi a New York, da Madrid a Londra. In ogni posto, mi son sempre sentito a casa: c'eri tu accanto a me. In questo grande viaggio, il posto accanto a me lo terrò sempre occupato per te. Ce lo siamo anche promesso il giorno del nostro matrimonio e farò del mio meglio per mantenere ogni singola promessa.

Voglio anche fermarmi un attimo e dirmi bravo. Sono passati anni da quando mi è stato detto non sarei mai diventato un ingegnere. Tanta fatica e tanta rabbia, ma anche tante soddisfazioni e tante gioie. Non ho mai avvertito avessi i favori dei pronostici, ma mi sono divertito a tirar fuori qualcosa che non pensavo di aver dentro. Sono cresciuto, ho sbagliato. Ho capito che non importa e mi sono perdonato per i miei sbagli. So che ogni decisione grande che ho preso, l'ho presa convinto fosse quella più giusta e tanto mi basta.
Non so se ho trovato il mio posto nel mondo, ma sicuramente ho capito come trovarlo: Ever failed. No matter. Try again. Fail again. Fail better.

Il Post Scriptum è per te, che stai per arrivare. Non esiste parola per descrivere quel che provo, ma in ogni momento in cui ho scritto questa tesi eri nei miei pensieri e, solo per questo, il periodo in cui ho scritto questo lavoro sarà uno dei ricordi più dolci della mia vita. Come ti dico ogni sera, cresci sana, cresci forte. Io non vedo l'ora di conoscerti e stringerti a me.

Contents

Chapter 1

Introduction and motivations

The goal of this work is to define and present a new formulation for a feedback control, and to show some of its potential applications in the field of mechanical engineering.
Control theory can be applied in everyday situations, from simple to complex tasks: the room temperature, the car velocity and the traffic flow stabilization [1], the blood pressure, he position of the flaps of an aircraft [2], the robot arms used in industrial manufacturing [3–7], the autonomous cars traffic flow [8–11], the quadcopter heading [12–16], and the list can become longer and longer. In fact, the power of control theory resides on its capability to be application-independent. The goal is to control a general situation called *system*, regardless what the system has inside: one simple example is the control of the human body temperature. The problem involves the blood circulation, the blood pressure, the external temperature, the heat loss mechanism and so on. A system can be defined in a wide range of disciplines and therefore control theory has a huge range of applications.
Control theory regards the altering of the future behaviour of a system, and to be successfully applied it needs three things:

1. An objective (purpose) that is linked with the future state of the system. It is straightforward the the past cannot be influenced and so the present.

2. A range of possible actions, so that some choices need to be done. If it is impossible to do any action, the system cannot be controlled.

3. In some sense, a choice of one of the possible actions to achieve the objective need to be done. This point requires a model, and the model has to produce a prediction of the action to the system state.

One of the main topics in control theory is how to choose the time horizon for the prediction resulting by actions. The longest the time horizon, the biggest the uncertainty is, and less accurate the control will be. If the corrective actions take effect relatively quickly, feedback control can be used. In fact, feedback controls are initiated by the error of the corrective action and their applications are typical engineering, such the control of the speed of a motor in the cruise control.
The innovative control here presented, named Feedback Local Optimality Principle (FLOP) has its mathematical basis in the optimal control theory, and specifically in

the variational approach for Optimal Control Problems. By a new proposition in the optimality condition, and by the acceptance of a local minimum instead of the global one achieved by classical theories, it succeeds in returning a feedback control law for a sufficiently wide class of dynamical systems. The FLOP control proved to be very flexible and robust, so it was applied in different fields of the mechanical engineering, from the obstacle avoidance for an autonomous ground vehicle to the control of a fleet of autonomous ships and to the control of a robot swarm.

Automatic control

Control theory has been developed to support the activity of automatic control. The main idea in automatic control is to replace the human worker with an automatic controller. The central idea behind the automatic control is the feedback control theory, which basis can be summed up as it follows:

1. Feedback control (or control loop) is a central idea of control theory. All control loops have the same basics form, and they are all represented in the same way.

2. All control loops are error driven, and the error is defined as the difference between the desired and the measured behaviours.

3. The rate of the error reduction represents an important performances measurement for the control system.

4. Control loops tend to become unstable if the requirement of the performances is too high.

In the simplest form of automatic control, the measurement of the system response y is tracked with the desired system response u. It is possible (and it is very common) that there is the presence of disturbing influences, w. This can be depicted as it follows:

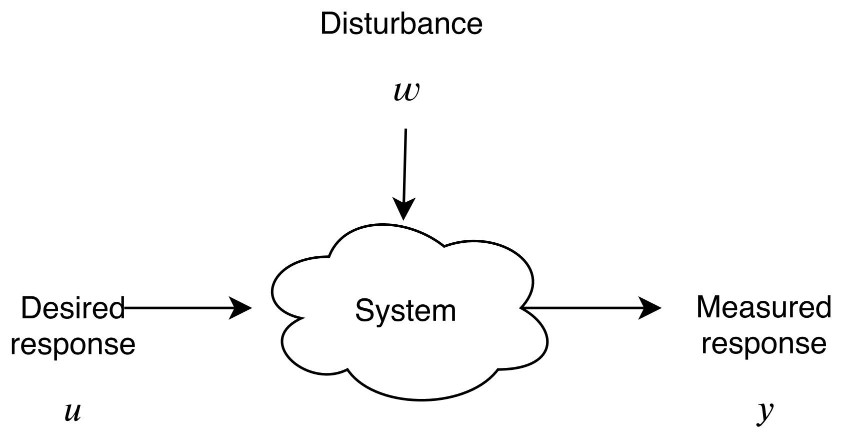

Figure 1.1. Automatic control system

One of the main questions which arises is: 'Can a realistic model be produced?' And even if a realistic model can be produced, can the model account also the presence of unknown disturbances? Automatic feedback control can positively answer to both questions, since it is an error-driven model. In the feedback automatic control, a device (controller) is connected to a process with the intention of influencing the behaviour of the process. The controller receives a continuous measurement of the difference between required and actual behaviours and its output is a to be determined function of this error.

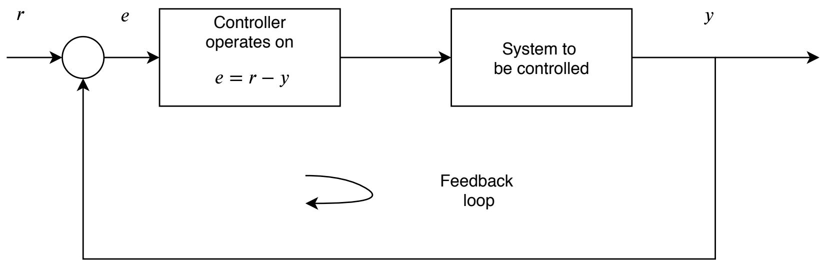

Figure 1.2. Automatic control loop

An alternative way to represent the system is with an artificial enhanced system, which later will be discussed in details. If the controller and the system are represented by two operators, G, D, respectively, we obtain:

$$\begin{aligned} \text{System output} &= Gu \\ \text{Controller output} &= De \\ \text{Controller input} &= e = r - y \end{aligned} \tag{1.1}$$

and it can be represented as it follows:

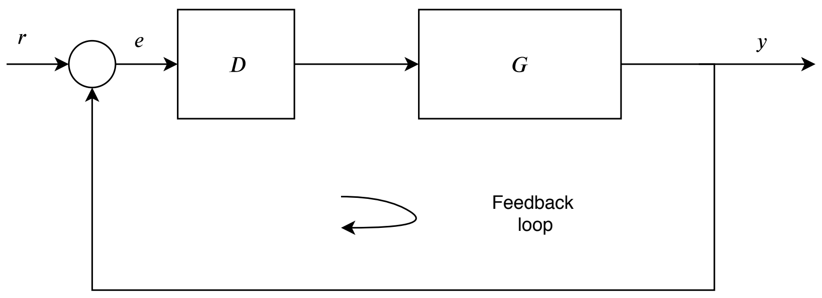

Figure 1.3. Artificial feedback control loop

By the nature of feedback control, corrective action begins once the error has been detected. Therefore, feedback control has the intrinsic request of having the rate of the corrective action at least equal to the rate of disturbance generation, and in some cases, this is not possible. Hence, feedback control has to be abandoned. Well-known and widely used alternatives to feedback control are:

- *Preprogrammed control*: a sequence of controls is evaluated in advance and it it implemented, no matter the disturbance nor the output from the process.
- *Feedforward control*: the disturbance signals are measured and eliminated before the error can occur.
- *Prediction followed by control*: a prediction of future conditions, based on predictive algorithms or known data on the past, are used to allow the best positioning of a low bandwidth control system.

A brief history of control theory

The history of control can go back in time until the antique Greece, one of the first works in the modern era is the one of James Watt on the centrifugal governor for the speed control of a steam engine, in the eighteen century. Other relevant early works were the Minorsky's [17] one on the control of a steering ship, the Nyquist's [18], frequency-response methods (especially Bode) lead to the possibility to design linear closed-loops control. In this period the Proportional-Integral-Derivative controller was developed and widely used for industrial control systems [19] [20]. During the 1950s the root-locus method was developed by Evans [21]. The frequency-response and root-locus methods lead to stable systems, satisfying some performance requirements. However, in any sense, these methods are considered as optimal. The problem of optimal systems arises around the end of the 1950s, and it regards in general multiple inputs, multiple output problems, while the classical control theory works mainly with a single input, single output problems. Around the 1960s, time-domain analysis using state variables have been developed [22]. From around 1960 to 1980 optimal control problem was widely treated, for both deterministic and stochastic methods [23–28]. Control theories used today can be divided into three main categories: classical control theory, modern control theory, and robust control theory.
The classical theory uses a mathematical model to define the relationship between the input and the output of the system. The most common type in this category is PID controllers [29–33]. The huge disadvantage of these techniques is they assume the system to be linear, or at least it must be linearized.
One of the prominent characteristics of modern control theory is the usage of the state-space model, which uses a set of first-order differential equations in the description of input-output system dynamics. The modern control theory is highly related to the concepts of modal control [34–38], optimal control based on the Pontryagin maximum principle [39, 40], dynamic programming [41–44].
Robust control usually deals with system analysis and control design fur imperfectly known process model. The robustness concept is in general in contraposition with the performance's one. To increase the robustness, in fact, in general, the control becomes less 'aggressive'.
Here, a short glossary is presented to give some crucial definitions used during this work:

- Dependent variable, or *output*. This variable must be controlled, so it must be made to behave in a prescribed way. For example, it may be necessary to assign the pressure or/and the temperature at a certain point during a process, or the velocity and the acceleration of a vehicle, the position of a particle and so on.

- Indipendent variable, or *input*. This variable is available to regulate and to control the behavior of the system. It can be a known data or it can be measured by some sensors.

- System: is a combination of components which act together and perform a certain objective. In general, it is not needed for the system to be physical. However, here the system will be intended as a physical one.

- Disturbance: is a type of signal which tends to affect the value of the output of a system in an adverse way. It is here intended a synonim of noise signals.

- Feedback control system: a system which is able to maintain a given relationship between the reference input and the output, by comparing them. One of the easiest example is given by a the control system for a room temperature: a desired temperature is set (controlled variable). The thermostat (control system) measures the temperature of the room and it increase or decrease the temperature of the room depending on the sign of the difference between the desired temperature and the actual one.

- Closed-Loop Control system: feedback control systems are often referred to as closed-loop control systems. In a closed-loop system, the difference between the input and the feedback signal (error) is given to the controller, which tries to set to zero the error.

- Feed-forward control system: the output has no effect on the control action. One example is the wash-machine: the machine works for a given period of time, regardless the cleanliness of the clothes.

The settings of the process can be summarized with the following figure:

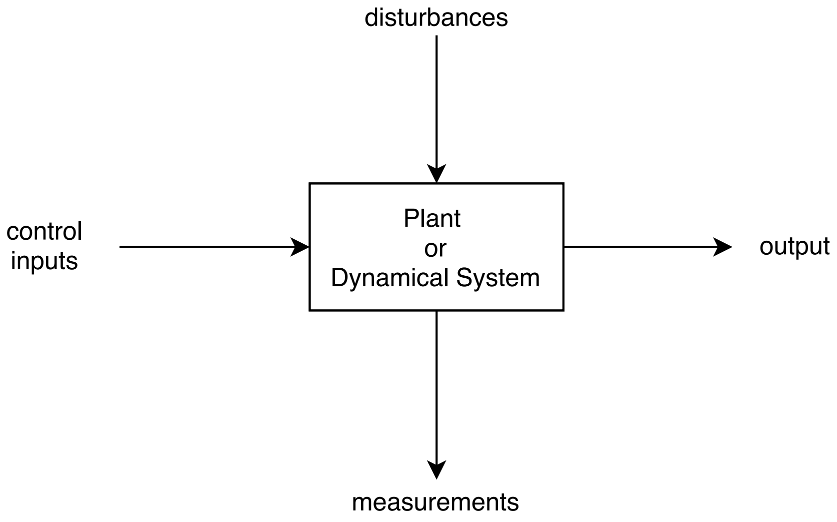

Figure 1.4. General Problem

In Figure 1.4 the input and the output arrows may represent both a scalar and vector inputs/outputs. Control is exercised by feedback which means the input is corrected by the available measurements. The controlled system can be represented by the feedback (closed-loop) system showed below. The aim of the control problem is to evaluate the characteristics of the controller so that the output can be:

1. Set to a defined value called reference.
2. Maintained at the chosen value despite the unknown disturbance.
3. Conditions (1) and (2) are satisfied, despite the uncertainties not only on the disturbance but also on the dynamical system modelization.

The above conditions are known as: tracking, disturbance rejection, robustness of the system.
The structure of this thesis is the following: after this introductory chapter, in chapter chapter 2 an overview of some of the most used engineering control techniques is given, together with the state of the art. Then, some optimal control techniques are described and main drawbacks of optimal control theory are pointed out, with the motivations for a new study in the field.
In chapter 3, the innovative feedback control named Feedback Local Optimality Principle (FLOP) is introduced, with an intuitive presentation whose purpose is to fix in the reader the idea behind the new control. Then, a rigorous mathematical approach is developed in detail, both for the mono-dimensional and the multi-dimensional cases.
In chapter 4 two applications of the FLOP strategy are shown. The goal of this chapter is to show some of the possibilities of the FLOP method with nonlinear dynamical systems. Here, some prototype problems are studied and compared with more classical techniques as the Linear Quadratic Regulator [45–47] and other Riccati-based techniques such as the State-Dependent Riccati Equation [48, 49].

In chapter 5, the swarm robotics in introduced as one of the most intriguing fields in modern engineering and the FLOP method is tested on different swarm robotics case to test the flexibility and the reliability of the developed control. In the last chapter, the conclusions of the work are delineated and the main results are highlighted.

Chapter 2

State of the art

This chapter will provide an overview of some of the most used control techniques in modern engineering. The purpose is to provide a wide outlook on nowadays used techniques, by describing their strengths and their weaknesses. Before introducing some commonly used techniques to solve the servomechanism problem, the basics concepts of the Laplace transform [50] are here highlighted, due its importance in control theory.

2.1 Laplace transform

The Laplace transform is a very useful tool in control theory. Therefore, its basics are here developed, following [51]. The Laplace transform of a function $f(t)$ is here denoted by $F(s)$, and its inverse is $f(t)$. The variable s a is complex one, whose role is defined by:

$$F(s) = \int_0^\infty exp(-st)f(t)dt \tag{2.1}$$

Example 1: $f(t) = k$, with k a constant. Then,

$$\begin{gathered} F(s) = \int_0^\infty exp(-st)k\,dt = \\ -\frac{1}{s}exp(-st)\bigg|_0^\infty = 0 - (-\frac{k}{s}) = \frac{k}{s} \end{gathered} \tag{2.2}$$

This is valid if the real part of the complex number s, $R(s)$ is positive, otherwise the integral does not exists. *Example 2*: $f(t) = \exp(at)$. Then,

$$\begin{gathered} F(s) = \int_0^\infty \exp(-st)\exp(at)\,dt = \\ -\frac{1}{a-s}\exp(a-s)t\bigg|_0^\infty = \frac{1}{a+s} \end{gathered} \tag{2.3}$$

This is true if $R(s) > a$ Starting from this, the usage of the Laplace transform in control theory is now highlighted:
a system receives an input $u(t)$ and it produces an output $y(t)$. The response is

determined by two factors: the nature of the input and the nature of the system. If the response of the system to a unite impulse applied at time $t = 0$ is $g(t)$, the response to any other input u is given by the convolution integral:

$$y(t) = \int_0^t g(t-\tau)u(\tau)d\tau \tag{2.4}$$

Convolution integrals are in general hard to solve, while the Laplace transform permits to solve the problem in a very simple way: $u(t), g(t)$ are transformed into $u(s), G(s)$ respectively, and it gets:

$$y(s) = G(s)u(s) \tag{2.5}$$

and this is possible since the transform-domain multiplication is equivalent to time-domain convolution. There is also another additional advantage in using the Laplace transform of the process, since the transformation back from transform-domain to time-domain it is often not necessary.

Transfer function

The transfer function of a dynamic system with $u(t), y(t)$ as input and output respectively, is defined to be the Laplace transform of the output under the condition that the input is a unit impulse applied at time $t = 0$. In practice:

$$G(s) = y(s)/u(s) \tag{2.6}$$

Equation (**??**) is valid for any u, y such that their transforms exist. If we consider two systems connected in series with responses $g_1(t), g_2(t)$, as depicted in the figure below:

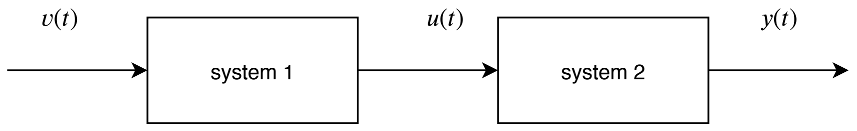

Figure 2.1. Two interconnected systems

Then,

$$\begin{aligned} y(t) &= \int_0^t g_2(t-\tau)u(\tau)d\tau \\ &= \int_0^t g_2(t-\tau)\int_0^\tau g_a(t-\rho)v(\rho)d\rho dt \end{aligned} \tag{2.7}$$

Eq (2.7) can be very difficult to solve, but using Laplace transformed signals and transfer functions, we obtain:

$$y(s) = G_2(s)G_1(s)v(s) \tag{2.8}$$

Through the Laplace transform and manipulation, block diagrams of any size and complexity can always be reduced to a single block by one of the following three rules depicted in Figure 2.2:

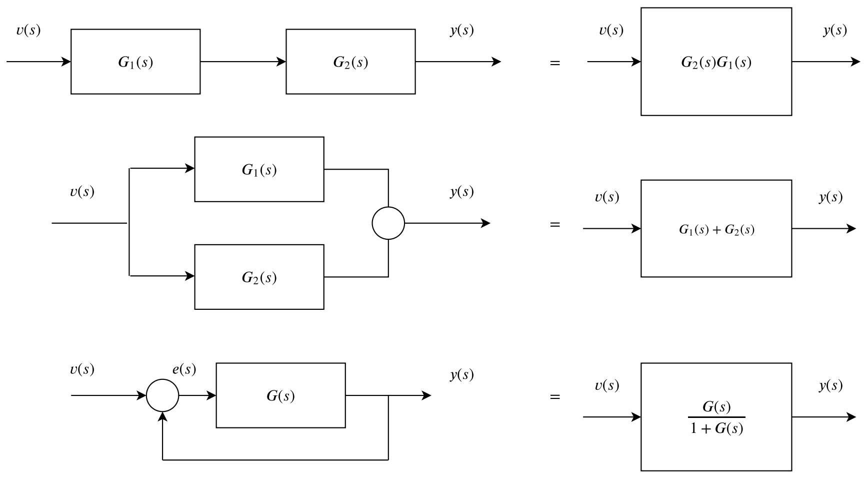

Figure 2.2. Basic configurations and equivalent single block representations

From differential equations to transfer function

A differential equation of the form

$$\frac{d^n y}{dt^n} + a_{n-1}\frac{d^{n-1}y}{dt^{n-1}} + \dots = b_r \frac{d^r u}{dt^r} + \dots$$

can be Laplace transformed to obtain:

$$\begin{aligned}(s^n + a_{n-1}s^{n-1} + \dots)y(s) &+ \text{terms depending on initial conditions}\\ = (b_r s^r + \dots)u(s) &+ \text{terms depending on initial conditions}\end{aligned}$$

Transfer function analysis assumes that initial condition effects have died away and that the output is a function of the input only. In that case, the transfer function corresponding with the differential equation is:

$$\frac{y(s)}{u(s)} = \frac{b_r s^r + \dots}{s^n + a_{n-1}s^{n-1}} \tag{2.9}$$

Note that this is valid for the transfer function analysis, not for the differential equation solution by Laplace transform.

Poles and zeros of a transfer function

Any value of the complex variable s for which $G(s) = 0$, is called a zero of $G(s)$. Any value p of s that satisfies $s \to 0 \implies G(s) \to \infty$ is called a pole of $G(s)$. If it is possible to write $G(s) = P(s)/Q(s)$, then the zeros are the roots of the equation $P(s) = 0$, while the poles are the roots of $Q(s) = 0$. The placement of zeros and poles is relevant in the understanding of the system behaviour. It is possible to represent poles and zeros in the complex plane, since the complex nature of s. In particular:

- the rate of change of transient solution increases as the pole distance to the origin increases;
- any pole in the right half of the plane indicates instability;
- poles on the real line indicate exponential, non-oscillatory responses;
- poles with non-zero imaginary parts indicate oscillatory solutions;
- poles nearest to the origin ('dominant poles') are most significant in governing the system behaviour.

Pole Placement

Let suppose a given system G has poles as shown in Figure 2.3, but it is required for the poles to be in the position of Figure 2.4. To do so, the system is preceded by an element D, which has the pole-zero diagram of Figure 2.5. This procedure will cancel the poles of G and will produce the required poles. This technique is known as pole-placement method. It is important to denote that the undesired poles of G are not removed, but they effect on the external behaviour is cancelled by the zeros of D. This method can leads to two possible difficulties:

1. Cancellation may not be exact or may not remains so, if it initially is.
2. The structure representing the cancelled terms is still present although it does not affect the behaviour of the system nor can be affected by outside events. The redundant structure can leads to anomalies and difficulties.

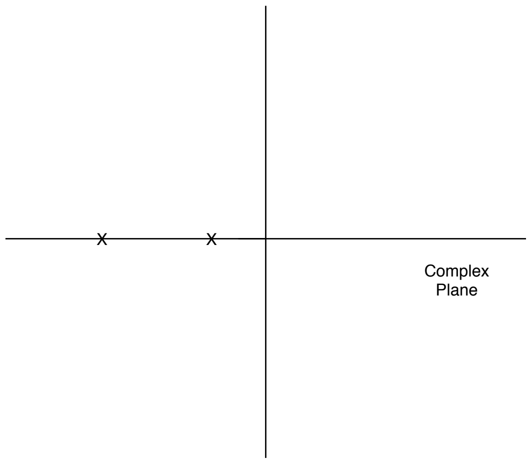

Figure 2.3. Original pole placement of the system

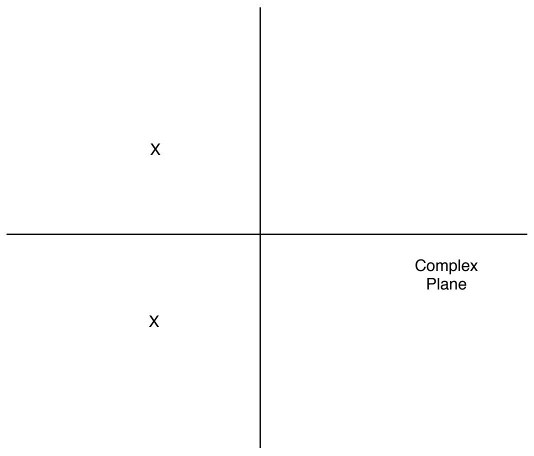

Figure 2.4. Desired pole placement of the system

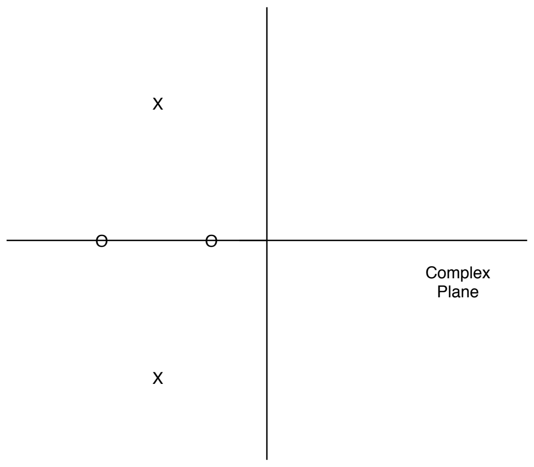

Figure 2.5. Poles and zeros of the controller to obtain the desired Pole Placement

The root locus technique

Consider the transfer function:

$$G(s) = \frac{C}{(s+1)(s+3)} \tag{2.10}$$

Poles are placed in $s = 1$ and $s = 3$. If the same system is connected in a closed loops, following the third rule in Figure 2.2, the overall transfer function for the configuration is:

$$\begin{aligned} \frac{G(s)}{1+G(s)} &= \frac{C}{(s+1)(s+3)} \Bigg/ \left(1 + \frac{C}{(s+1)(s+3)}\right) \\ &= \frac{C}{(s+1)(s+3)+C} \end{aligned} \tag{2.11}$$

Poles of the closed loop system are found by equating the denominator of the transfer function to zero, therefore:

$$(s+1)(s+3)+C=0$$

The solutions are: $s=-2\pm\sqrt{1-C}$. Hence, for:

- $C<1$ the poles are real, unequal
- $C=1$ the poles are real, equal
- $C>1$ the poles are complex conjugates.

The above brief discussion on the Laplace transform is useful mostly for the next section, aimed to the Proportional-Integral-Derivative Controllers, whome most results are strictly related to the Laplace transform.

2.2 PID Controllers

Here, the important class of controllers known as three term controllers is discussed. In particular, the Proportional-Integral-Derivative (PID) controller is considered. This kind of controller has three adjustable parameters and is widely used in the engineering disciplines. This class of controller is applied to the servomechanism problem [52] , which in the most simple case is a three blocks problem, as represented in Figure 2.6

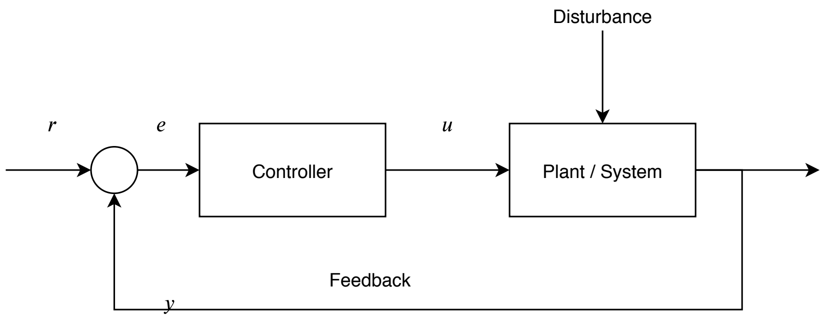

Figure 2.6. Simplest case of servomechanism

There is no such thing as a recognized definition of a servomechanism (or simply *servo*), but it can be considerable as a servo any electrical, electronic, mechanical, hydraulic system which uses feedback. First of all, a servo is a control device, whose job is primarily to position a load. The load can be any kind of load, from a carriage to an hydraulic piston, to an electric motor. The input to a servo is in general a small signal, but more generally is the error between a desired load and the actual measured load. The servo is considered a closed loop system, because it uses feedback.

Integral Control

As a starting point, let consider one of the simplest possible controls: the Integral Control (IC) IC is used in very different applications in the control industry, especially to design robust servomechanisms. The input-output relationship is

$$y(t) = K \int_0^t u(\tau)d\tau + y(0) \tag{2.12}$$

which can be also written in differential form:

$$\frac{dy(t)}{dt} = Ku(t) \tag{2.13}$$

where K is the integrator gain. If the output $y(t)$ is a constant, from (2.13) follows that:

$$\frac{dy(t)}{dt} = 0 = Ku(t)\ \forall\ t > 0 \tag{2.14}$$

Equation (2.14) proves two facts about the IC:

1. If the output of an IC is constant over a chosen time interval, then the input must be equal to zero over that same interval.
2. If the input is nonzero, the output changes.

These two points clarifies how to use the IC in the servomechanism problem: if the dynamic system requires to track a constant reference r, it is enough to attach the integrator to the dynamical system and evaluate the error

$$e(t) = r - y(t)$$

as the input to the integrator. Moreover, it is crucial to ensure the closed-loop system is asymptotically stable so that under constant reference (and disturbance) all signals reach constant steady-state values.

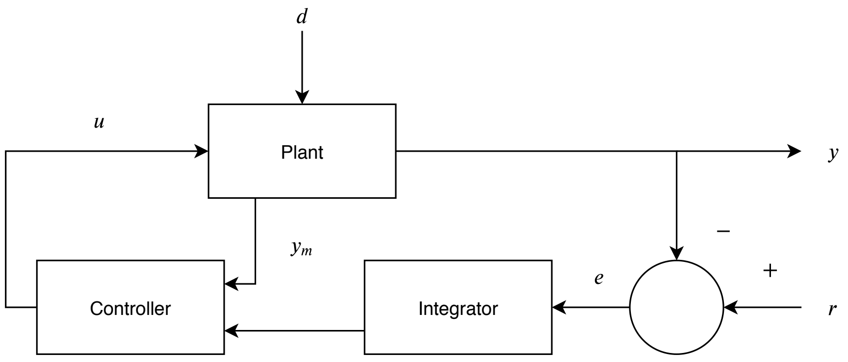

Figure 2.7. Schematic of the servomechanism scheme

In the case depicted in Figure 2.7, the integrator output $v(t)$ tends to a constant value and, as a consequence, the input tends to zero. Since the input is imposed as the tracking error, this means that $e(t) = r - y(t) \to 0$ when $t \to \infty$. It is important to note that, the steady-state tracking property here discussed is robusts. In fact, it holds as long as the closed loop is asymptotically stable and it is:

1. independent of the initial conditions of the system and the controller
2. independent of the constant disturbances/references
3. independent of the linearity/nonlinearity of the dynamical system.

In these conditions, the tracking problem becomes a more simple stability problem. If the stability is assured, the result is assured as well.

PID Controllers

In the previous subsection was pointed out that when an IC is part of a stable (asymptotically) system and a constant input is applied to the system, the integrator input is forced to zero. This principle is the base for the design of servomechanisms: they can be linear, nonlinear, single-input single-output, and multivariable. Starting from the IC, free parameters for stabilization can be added, without influencing the zeroing property, by adding parallel branches to the controller. In these branches, the error itself and its derivative are added to the integral of the error. This leads to the Proportional-Integrative-Derivative controller, or PID.

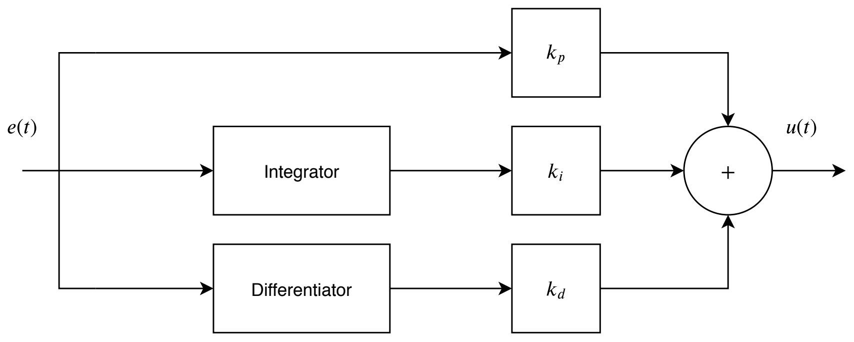

Figure 2.8. Schematic of the PID scheme

In Figure2.8, three different parameters are shown and they represent the proportional, integral and derivative gains, k_p, k_i, k_d respectively. If the closed loop is stable, the integrator input will go to zero, no matter the gains value. Hence, the purpose of the gains is to stabilize the closed-loop system (if possible) and to increase the performances in terms of robustness and transient response. As guide line, the derivative is useful if the error is varying slowly. In fact, the derivative term can be omitted if high-frequency noise is present in the error signal. In the case of high-frequency noise, the controller usually becomes a Proportional-Integral one, or PI. The PID controller is, as it has been said, widely used in today's industry. Since its importance, many different design strategies were presented over the years. Here, a quick outlook of some classical designs [19] for the PID controller are discussed:

- The Ziegler-Nichols step response method
- Internal Model Controller
- Dominant Pole Design: The Cohen-Coon method

Ziegler-Nichols step response method

This method is based on the criterion for which the control system has fast control with satisfactory stability. In general, these two criteria are contradictory: a very good stability corresponds to a slow control, while poor stability corresponds to fast control. Clearly, both a slow control and a non-stable control are not desirable in control theory. A good tuning method is able to compromise these two requirements and to obtain the fastest control with a satisfactory stability. With satisfactory stability, here is intended that the output response converges to a constant value with satisfactory damping after a time-limited change of the setpoint or the disturbance. It can be quantified in several ways. The Ziegler-Nichols method states a satisfactory damping corresponds to an amplitude ratio around 0.25 between subsequent peaks. Despite the work was published in early '40s and the huge research on the PID controllers since then, the Ziegler-Nichols method is still widely used. The PID controller is implemented in the Laplace domain as follows:

$$C(s) = k_p + \frac{k_i}{s} + k_d s \tag{2.15}$$

The derivative term is often replaced by

$$\frac{k_d s}{1 + T_d s}$$

where T_d is small, positive, and in general fixed. This method is an experimental open-loop tuning method and it can be applied to open-loop stable systems. This methods first individuates two parameters depending on the step response of the system. These parameters are determined as follows:

1. The point of maximum slope due the step response is determined, and the tangent to the point is drawn.
2. The intersection between the tangent with the vertical and the horizontal axes give A, L respectively.
3. The PID parameters are the following:

$$k_p = \frac{1.2}{A} \quad k_i = \frac{0.6}{AL} \quad k_p = \frac{0.6L}{A}$$

From experimental results can be seen that these setting rules for the controller parameters led the system to obtain an amplitude decay ratio of 0.25, which means the first overshoot decays to a quarter of its original value after just one oscillation.

Internal Model Controller

The Internal Model Controller (IMC) idea is based on the Internal Model Principle, which states that *control can be achieved only if the control system encapsulates, either implicitly or explicitly, some representation of the process to be controlled* [20] This means that, if the control has been found on the exact model of the process, the perfect control is theoretically possible. However, process-model mismatch is usually common, for example for some unknown disturbance in the process. The strategy has the general structure depicted in Figure 2.9:

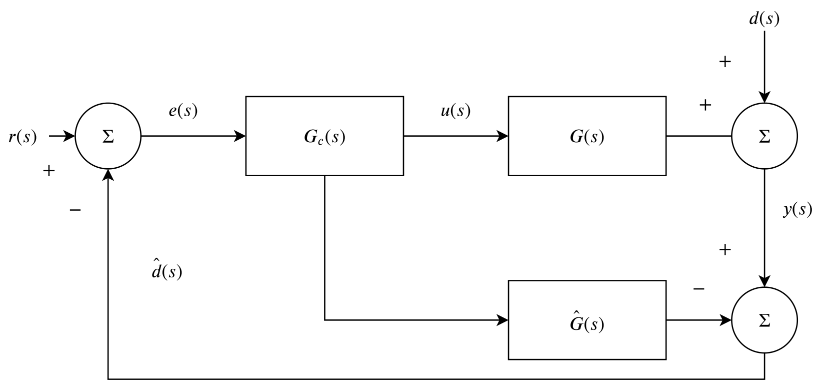

Figure 2.9. Schematic of the IMC scheme

In the diagram, $d(s)$ is the unknown disturbance. The input $u(s)$ is applied both on the real process and its model, $G(s)$ and $\hat{G}(s)$ respectively, while $G_c(s)$ is the controller. The real process output, $y(s)$, is compared with the model output, giving $\widehat{d(s)}$. So:

$$\hat{d}(s) = \Big[G(s) - \hat{G}(s)\Big]u(s) + d(s)$$

and it represents the information that is missing in the model. Therefore, $\hat{d}(s)$ can be used to improve control. This is done by subtracting $\hat{d}(s)$ to the reference $r(s)$, and the resulting control signal becomes:

$$u(s) = \Big[r(s) - \hat{d}(s)\Big]G_c(s) = \Bigg[r(s) - \Big[G(s) - \hat{G}(s)\Big]u(s) - d(s)\Bigg]G_c(s)$$

Since $y(s) = G(s)u(s) + d(s)$, by rewriting in a more practical way the control signal $u(s)$, the closed loop transfer function is therefore:

$$y(s) = \frac{\Big[r(s) - d(s)\Big]G_c(s)G(s)}{1 + \Big[G(s) - \hat{G}(s)\Big]G_c(s)} + d(s) \tag{2.16}$$

Once the general idea is given, this can be applied on a PID controller: if $G_c(s)$ is a PID, the diagram of the process can be represented by Figure 2.10

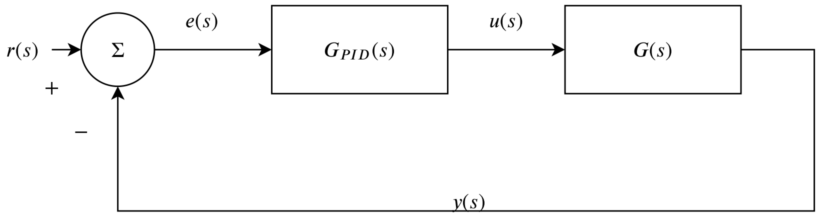

Figure 2.10. Schematic of the PID - IMC scheme

and

$$G_{PID}(s) = \frac{G_{IMC}(s)}{1 - G_{IMC}(s)\hat{G}(s)}$$

Assuming:

$$G(s) = \frac{k}{1+Ts}e^{-Ls} \tag{2.17}$$

and by use the Padè approximation, so that:

$$e^{-Ls} \approx \frac{1-\frac{L}{2}s}{1+\frac{L}{2}s}$$

The model is given by:

$$G(s) = \frac{k}{1+Ts}\frac{1-\frac{L}{2}s}{1+\frac{L}{2}s}$$

and after some calculations the following parameters for a standard PID can be evaluated:

$$k_p = \frac{2T+L}{2k(L+\lambda)}$$
$$k_i = \frac{1}{k(L+\lambda)}$$
$$k_d = \frac{TL}{2k(L+\lambda)}$$

whre λ is assumed to be small. In the work [CITE Morari e Zafiriou] a suitable choice is considered to be $\lambda > 0.2\,T$ and $\lambda >\ 0.25L$.

Dominant Pole Design: The Cohen-Coon method

The dominant pole design try to place a few poles to give some control performance specifications. Starting from (2.17), the method try to obtain the 0.25 amplitude decay for load disturbance response by placing one real pole and two pair of complex poles. With this approach, the integrated error in the time-interval $t \in [0, \infty)$ is minimized. Based on both analytical and numerical computation, the PID controller parameters obtained by Cohen and Coon provides:

$$k_p = \frac{1.35(1-0.82b)}{a(1-b)}$$
$$k_i = \frac{1.35(1-0.82b)(1-0.39b)}{aL(1-b)(2.5-2b)}$$
$$k_d = \frac{1.35L(0.37-0.37b)}{a(1-b)}$$

with:

$$a = \frac{kL}{T},\ \ b = \frac{L}{L+T}$$

It is interesting to denote that, for small b, the controller parameters are similar to the ones obtained by the Ziegler-Nichols method.

2.3 Optimal control

In the previous section a brief review on classical control system design is given. In general, is a so-called trial-and-error process in which different strategies of analysis are used iteratively to design an 'acceptable' system. One of the main issues is to define what is considerable as 'acceptable', but it is in general defined in terms of time and/or frequency domain criteria (rise time, settling time, peak overshoot, gain, bandwidth, and so on). However, in the case of multi-input or multi-output systems it is possible that completely different performance criteria must be satisfied. For instance, it is impossible to solve the problem of the minimization of the fuel expenditure of a racing car through classical theory. A different approach for these problems is now presented, and it goes under the name of optimal control theory [23, 53]. The objective of optimal control theory is the determination of the control signals that will permit to a process to satisfy the physical constraints, and at the same time will maximize (or minimize) some performance criterion. Since it is crucial to have a well-defined problem, in this first part of the section some aspects on the formulation, the notation and nomenclature are here introduced. The formulation of an optimal control problem requires:

1. A mathematical description of the process to be controlled ;
2. A statement of the physical constraints;
3. Specification of a performance criterion;

The mathematical model

In any control problem a nontrivial part is the process modeling. The goal is to find the simplest mathematical description that describes and predicts the response of the physical system to all inputs. Introducing the state variable (or state) of the process at time t as:

$$x_1(t), x_2(t), \ldots, x_n(t),$$

and the control inputs as:

$$u_1(t), u_2(t), \ldots, u_m(t),$$

the system may be described by n first order differential equations:

$$\begin{aligned}
\dot{x}_1(t) =& f_1\Big(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), u_2(t), \ldots, u_m(t), t\Big) \\
\dot{x}_2(t) =& f_2\Big(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), u_2(t), \ldots, u_m(t), t\Big) \\
&\cdot \\
&\cdot \\
&\cdot \\
\dot{x}_n(t) =& f_n\Big(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), u_2(t), \ldots, u_m(t), t\Big)
\end{aligned} \tag{2.18}$$

We can define the *state vector* of the system as:

$$\boldsymbol{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$$

and

$$\boldsymbol{u}(t) = [u_1(t), u_2(t), \dots, u_m(t)]^T$$

as the control vector, so that the state equation (2.18) can be written as:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\Big(\boldsymbol{x}(t), \boldsymbol{u}(t), t\Big) \tag{2.19}$$

with $\boldsymbol{f}$ is defined as it follows directly from (2.18).

Physical Constraints

After the selection of the mathematical model, physical constraints have to be defined both on the state and control values. It is possible to have some time-position constraint, i.e. the object has to be in a certain position x_0 in a given instant t_0. If some constraints exist both on the initial and final time of the process, they are called boundary conditions. Even in the control variable some constraints are possible, for instance if the control represent the acceleration and the breaking capability of a car, the constraints are represented by the maximum acceleration and deceleration of the car. Moreover, some global properties can lead to other constraints: if from initial to final time is impossible to refuel the car, the initial amount of car is an upper boundary for the maximum fuel the car can use in the selected time interval. In a more precise way, a control history able to satisfy the control constraints during the selected time interval $[t_0, t_f]$ is called *admissible control*. Here, the set of admissible control will be represented by U, and $u \in U$ means that u is admissible.
With a very similar definition, it can be introduced the *admissible trajectory*, the state trajectory able to satisfy the state variable constraints during the entire time interval $[t_0, t_f]$. The set of admissible trajectories is represented by X and $x \in X$ means the trajectory is admissible. In general, the final state of a system is required to lie in a specified region T, named the *target set* or target. If the final state and the final time are chosen, then T is a point, x_T.

The Performance Measure

In optimal control theory, the performance of the system are evaluated quantitatively. To do so, the designer of the system chooses a performance measure. On the one hand, in some cases the performance index is clearly stated in the problem, i.e. 'Transfer the object from A to B as quickly as possible'. On the other hand, if the problem is 'Maintain the position and the velocity of the system near zero with a small expenditure of control energy'. While in the first case it is clear what the performance index should be (e.g. the elapsed time), in the second case it is not really clear. In such problems it is possible that multiple performance indexes have to be tried, before selecting the one which leads to the optimal performance. In the general form, the performance index will be evaluated by a measure of the form:

$$J(\boldsymbol{u}) = \int_{t_0}^{t_f} \mathcal{L}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)dt \tag{2.20}$$

where t_0, t_f are the initial and final time and $\mathcal{L}$ is the Lagrangian. The Lagrangian is assumed to be defined and continuous, together with its partial derivatives $\mathcal{L}_x$ on

$R \times R^{n_x} \times R^{n_u}$. Usually, the initial state is known. So, starting from $\boldsymbol{x}(t_0) = \boldsymbol{x_0}$, applying the control $\boldsymbol{u}(t)$ for $t \in [t_0, t_f]$, the system will follow a state trajectory. The performance measurement J assigns to the trajectory a real number. For different trajectories, different numbers are given.

The Optimality Criteria

The statement of the optimal control problem can be expressed as it follows: find an *admissible control* $\boldsymbol{u}^*$ which causes the system

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \tag{2.21}$$

to follow an *admissible trajectory* $\boldsymbol{x}^*$ that *minimizes* the performance measure

$$J(\boldsymbol{u}) = \int_{t_0}^{t_f} \mathcal{L}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)dt \tag{2.22}$$

$\boldsymbol{u}^*$ is then called the *optimal control* and $\boldsymbol{x}^*$ is the *optimal trajectory.* Given the above definition, some comments can be made:

1. Sometimes it is possible that there is no information about the existence of an optimal control.

2. Even if the optimal control exists, it may be not unique.

3. If $\boldsymbol{u}^*$ minimized the performance measure to be minimized, it means:

$$\begin{aligned} J^* = J(\boldsymbol{u}^*) &= \int_{t_0}^{t_f} \mathcal{L}(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t), t)dt \\ &\leq \int_{t_0}^{t_f} \mathcal{L}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)dt \\ &\forall\, \boldsymbol{u} \in U \text{ s.t. } \boldsymbol{x} \in X \end{aligned} \tag{2.23}$$

The first point leads to the conclusion that it may not exists an admissible control which causes the system to move in an admissible trajectory. In general problems, there are not many existence theorems, so it is possible that is simpler to try to find an optimal control rather then try to prove that one optimal control exists. Nonunique optimal controls may be helpful, in the sense the multiple configurations can lead to the same performance measure and so the possibility to choose among several controller configurations. However, nonunique optimal control have the disadvantage of computational procedures, which can become longer. The inequality (2.23) states that the performance measure must have a value smaller (or equal) than any other value generated by an admissible control and trajectory. So, the purpose is to find the absolute (global) minima, not just a local one.

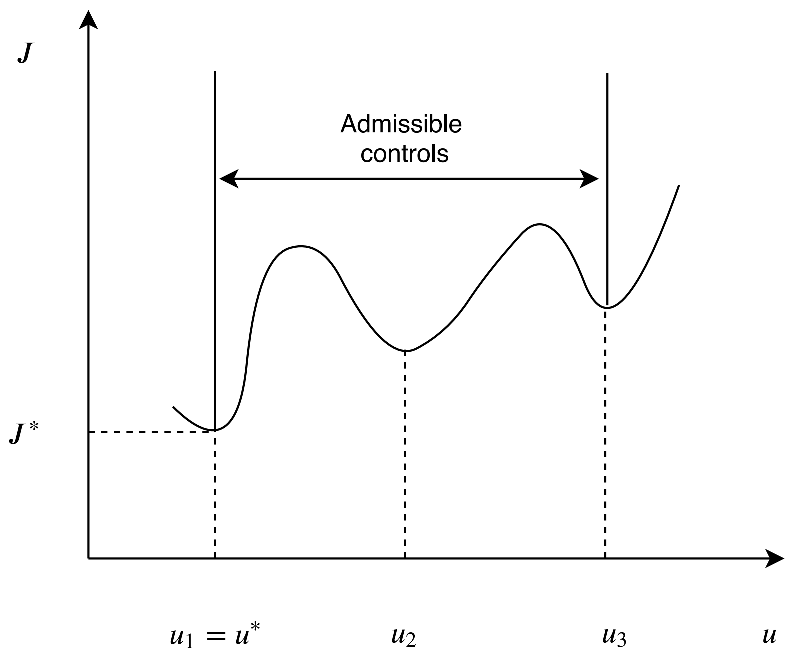

Figure 2.11. Representation of global minimum

Performance measures for typical optimal control problems

Here, some of the typical optimal control problems are discussed, according to [23]:

Minimum-time problems

Statement: transfer a system from the initial state $\boldsymbol{x}(0) = \boldsymbol{x}_0$ to the state target $\boldsymbol{x}_T$ in the minimum amount of time. The performance index is:

$$J = t_f - t_0 = \int_{t_0}^{t_f} dt \tag{2.24}$$

Terminal control problems

Statement: minimize the deviation of the final state of the system $\boldsymbol{x}(t_f)$ from its desired value $\boldsymbol{r}(t_f)$. Using matrix notation, this can be written as

$$J = [\boldsymbol{x}(t_f) - \boldsymbol{r}(t_f)]^T \boldsymbol{Q} [\boldsymbol{x}(t_f) - \boldsymbol{r}(t_f)] \tag{2.25}$$

where $\boldsymbol{Q}$ is a real symmetric positive semi-definite $n \times n$ weighting matrix.

Minimum-control effort problems

Statement: transfer a system from the initial state $\boldsymbol{x}(0) = \boldsymbol{x}_0$ to the state target $\boldsymbol{x}_T$ with the minimum control effort. The performance index is:

$$J = \int_{t_0}^{t_f} \boldsymbol{u}(t)^T \boldsymbol{R} \boldsymbol{u}(t) \tag{2.26}$$

where $\boldsymbol{R}$ is a real symmetric positive definite weighting matrix.

Tracking problems

Statement: maintain the system state $\boldsymbol{x}(t)$ as close as possible to the desired trajectory state $\boldsymbol{x}_T(t)$ in the entire time interval $[t_0, t_f]$. The performance index is:

$$J = \int_{t_0}^{t_f} [\boldsymbol{x}(t) - \boldsymbol{x}_T(t)]^T \boldsymbol{Q} [\boldsymbol{x}(t) - \boldsymbol{x}_T(t)] dt \tag{2.27}$$

Open-Loop Vs Closed-Loop Optimal Control

One of the ultimate goals in the optimal control theory is to synthesize an optimal control law, which can be used at any time t and for any feasible state value at t. **Closed-loop Optimal Control**: if a functional relation of the form

$$\boldsymbol{u}^*(t) = \boldsymbol{\omega}(t, \boldsymbol{x}(t)) \tag{2.28}$$

can be found for the optimal control at time t, then $\boldsymbol{\omega}$ is named the closed-loop (feedback) optimal control for the problem. For instance, in the case of linear ODE systems, under certain assumption (quite general) it is possible to obtain an optimal control in the form of (2.28), and the feedback control form is linear, time-varying:

$$\boldsymbol{u}^* = -\boldsymbol{K}(t), \boldsymbol{x}(t) \tag{2.29}$$

Despite some well-known cases, obtaining an open-loop optimal control law is much easier from a practical viewpoint. **Open-Loop Optimal Control**: if the optimal control law is determined as a function of time for a specified initial state value, in the form:

$$\boldsymbol{u}^*(t) = \boldsymbol{\omega}(t, \boldsymbol{x}(t_0)) \tag{2.30}$$

the optimal control is said to be in open-loop form. Therefore, an open-loop optimal control is optimal only for a specified initial state value. In the literature, usually open-loop optimal control is also referred to as dynamic optimization problems. Reasonably, open-loop optimal controls are rarely applied in practice due the presence of uncertainty (e.g. noise, model mismatch, variation of the initial condition). Uncertainties may lead the system to operate in sub-optimality condition or, worst case scenario, can lead to infeasible operation due to constraints violation. Nevertheless, the knowledge of an optimal open-loop (feed-forward) optimal control for a known process can provide some insights on how to improve system operations. Moreover, open-loop optimal controls are widely used in a number of feedback control algorithms which go under the definition of Model Predictive Controls (MPCs). Optimal control problems encompass problems of the calculus of variations and, therefore, some difficulties arises regarding the existence of a solution. Apart from the case where no admissible control exists for a problem, the absence of an optimal control lies to the fact that many controls fail to be compact.

The Calculus of Variations

Optimization problems can be solved using the branch of mathematics known as calculus of variations. As it has been said in the previous section, in optimal control problems the objective is to determine a function that minimizes a specified functional, i.e. the performance measure. The analogous problem in calculus is to determine a point that yields the minimum value of a function [53]. In calculus of variations, the definition of a functional J is similar to the definition of a *function*:

- A function f is a rule of correspondence that assigns to each element $\boldsymbol{q}$ in a given set $\mathcal{D}$ a unique element in a set $\mathcal{R}$. Here, $\mathcal{D}, \mathcal{R}$ are called the *domain* of f and its *range*.
- A functional J is a rule of correspondence that assign to each function $\boldsymbol{x}$ in a certain class Ω a unique real number. Ω is here called domain of the functional, and the set of possible real numbers associated is its range.

Now, some definition and properties for a generic functional J are here given:

1. **Linearity**: J is a *linear functional* of $\boldsymbol{x}$ if and only if it satisfies two principles, the homogeneity and the additivity ones. The principle of homogeneity states that

$$\begin{gathered} J(\alpha \boldsymbol{x}) = \alpha J(\boldsymbol{x}), \\ \forall\, \boldsymbol{x} \in \Omega \text{ and } \forall\, \alpha \in R : \alpha \boldsymbol{x} \in \Omega \end{gathered} \tag{2.31}$$

   while the principle of additivity states:

$$\begin{gathered} J(\boldsymbol{x}^{(1)} + \boldsymbol{x}^{(2)}) = J(\boldsymbol{x}^{(1)}) + \boldsymbol{x}^{(2)} \\ \forall\, \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \text{and } \boldsymbol{x}^{(1)} + \boldsymbol{x}^{(2)} \in \Omega \end{gathered} \tag{2.32}$$

2. **Closeness**: If two points are said to be close to one another, a geometric interpretation come easily to mind. The definition of two *functions* close to one another is less immediate. To define the closeness between functions, the concept of norm shall be introduced: The norm of a function is a rule of correspondence that assign to each function $\boldsymbol{x} \in \Omega$ defined for $t \in [t_0, t_f]$ a real number. The norm of $\boldsymbol{x}$, denoted by $||\boldsymbol{x}||$, satisfies the following properties:
   - $||\boldsymbol{x}|| \geq 0$ and $||\boldsymbol{x}|| = 0 \iff \boldsymbol{x}(t) = \mathbf{0}\; \forall\, t \in [t_0, t_f]$
   - $||\alpha \boldsymbol{x}|| = |\alpha| \cdot ||\boldsymbol{x}||\; \forall\, \alpha \in R$
   - $||\boldsymbol{x}^{(1)} + \boldsymbol{x}^{(2)}|| \leq ||\boldsymbol{x}^{(1)}|| + ||\boldsymbol{x}^{(2)}||$

3. **Increment of a functional**: if $\boldsymbol{x}$ and $\boldsymbol{x} + \delta \boldsymbol{x}$ are functions for which the functional J is defined, the increment of J, denoted by ΔJ is:

$$\Delta J = J(\boldsymbol{x} + \delta \boldsymbol{x}) - J(\boldsymbol{x})$$

   In general, it can be written $\Delta J(\boldsymbol{x}, \delta \boldsymbol{x})$ and $\delta \boldsymbol{x}$ is the *variation* of the function $\boldsymbol{x}$.

4. **Variation of a functional** can be written as

$$\Delta J(\boldsymbol{x}, \boldsymbol{\delta x}) = \delta J(\boldsymbol{x}, \boldsymbol{\delta x}) + g(\boldsymbol{x}, \delta \boldsymbol{x}) \cdot ||\delta \boldsymbol{x}||$$

   where δJ is linear in $\delta \boldsymbol{x}$. If

$$\lim_{\delta \boldsymbol{x} \to 0} g(\boldsymbol{x}, \delta \boldsymbol{x}) = 0,$$

   J is said to be differentiable on $\boldsymbol{x}$ and δJ is the variation of J evaluated for the function $\boldsymbol{x}$.

5. **Maxima and minima of a functional**. A functional J with domain Ω has a relative extremum at $\boldsymbol{x}^x$ if there is an $\varepsilon > 0 \mid \forall\, \boldsymbol{x} \in \Omega$ which satisfy $||\boldsymbol{x} - \boldsymbol{x}^*|| \leq \varepsilon$. If
$$\Delta J = J(\boldsymbol{x}) - J(\boldsymbol{x}^*) \geq 0,$$
$J(\boldsymbol{x}^*)$ is a relative minimum, if
$$\Delta J = J(\boldsymbol{x}) - J(\boldsymbol{x}^*) \leq 0,$$
$J(\boldsymbol{x}^*)$ is a relative maximum. $\boldsymbol{x}^*$ is an extremal and $J(\boldsymbol{x}^*)$ is an extremum.

6. **The fundamental theorem of calculus of variations** Let $\boldsymbol{x}$ be a vector function of t in $\boldsymbol{\Omega}$, and $J(\boldsymbol{x})$ be a differentiable functional of $\boldsymbol{x}$. Assume that the functions in $\boldsymbol{\Omega}$ are not contrained by any boundaries. The theorem states that: if $\boldsymbol{x}^*$ is an extremal, the variation of J must be vanish on $\boldsymbol{x}^*$, that is:
$$\delta J(\boldsymbol{x}^*, \delta \boldsymbol{x}) = 0 \,\forall\, \text{admissible}\ \delta \boldsymbol{x} \tag{2.33}$$

2.3.1 Variational approach for Optimal Control

Even though some sufficient conditions can be discussed, they are not useful at all in helping to find solutions. Although, here some conditions which any optimal control must satisfy can be discussed. The remarkable thing of these condition is that often they allow to single out a small set of controls, sometimes even a single control, so if an optimal control exists, it is possible that it can be found in these candidates. However, it is here emphasized that necessary conditions may give in result a non-empty set of candidates, but the optimal control could not exist anyway, for that problem.

Euler-Lagrange Equations

In calculus of variation, the simplest problem has both endpoints fixed. In optimal control, however, the simplest problem involves a free value of the state variables at the terminal point, and the problem is usually written [53]:
$$\begin{gathered} \min J(\boldsymbol{u}) = \int_{t_0}^{t_f} \mathcal{L}(t, \boldsymbol{x}(t), \boldsymbol{u}(t))dt \\ \text{subject to: } \dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t));\ \boldsymbol{x}(t_0) = \boldsymbol{x_0} \end{gathered} \tag{2.34}$$

Initial and final time are fixed. Since once the control function $\boldsymbol{u}$ is found, it determines the response $\boldsymbol{x}$ together with the initial condition (if it exists), the purpose of this approach is to find the control.
Here, the first-order necessary conditions are developed. The problem to be considered is to minimize the functional $J(\boldsymbol{u})$ in (2.34). Control variable is assumed $\boldsymbol{u} \in \mathcal{C}[t_0, t_f]^{n_u}$, endpoints are fixed and $t_0 < t_f$, $\mathcal{L}$ and $\boldsymbol{f}$ are continuous in $(t, \boldsymbol{x}, \boldsymbol{u})$ and have continuous first partial derivatives with respect to $\boldsymbol{x}, \boldsymbol{u}\ \forall\ (t, \boldsymbol{x}, \boldsymbol{u}) \in [t_0, t_f] \times R^{n_x} \times R^{n_u}$.Supposing $\boldsymbol{u}^* \in \mathcal{C}[t_0, t_f]^{n_u}$ to be a local minimizer for the problem, and letting $\boldsymbol{x}^* \in \mathcal{C}[t_0, t_f]^{n_x}$ to be the denoted response,

then there is a vector $\boldsymbol{\lambda}^* \in \mathcal{C}[t_0, t_f]^{n_x}$ such that the triple $\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*$ satisfies the following system:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}) \\ \dot{\boldsymbol{\lambda}} &= -\mathcal{L}_{\boldsymbol{x}}(t, \boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}_{\boldsymbol{x}}(t, \boldsymbol{x}, \boldsymbol{u})^T \boldsymbol{\lambda} \\ 0 &= -\mathcal{L}_{\boldsymbol{u}}(t, \boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{f}_{\boldsymbol{u}}(t, \boldsymbol{x}, \boldsymbol{u})^T \boldsymbol{\lambda} \\ \text{with } \boldsymbol{x}(t_0) &= \boldsymbol{x_0}, \ \boldsymbol{\lambda}(t_f) = \boldsymbol{0} \end{aligned} \tag{2.35}$$

These equations are known as the Euler-Lagrange equations and the second one is often referred as the adjoint (costate) equation. The third set of equations in (2.35) consist of n_u algebraic equations. The first two lines represent $2 \times n_x$ ODEs and their respective boundary conditions. Therefore, the Euler-Lagrange equations provide a complete set of necessary conditions, but some boundary conditions are given at the initial time (t_0) and others at the final time (t_f). These kind of problems are usually known as two-point boundary value problems. In this formulation is sometimes useful to introduce the Hamiltonian formulation $\mathcal{H}$, defined as:

$$\mathcal{H} = \mathcal{L}(t, \boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^T \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}) \tag{2.36}$$

So that Euler-Lagrange equations in (2.35) can be written as:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \mathcal{H}_{\boldsymbol{\lambda}} \quad \boldsymbol{x}(t_0) = \boldsymbol{x_0} \\ \dot{\boldsymbol{\lambda}} &= \mathcal{H}_{\boldsymbol{x}} \quad \boldsymbol{x}(t_f) = \boldsymbol{0} \\ \boldsymbol{0} &= \mathcal{H}_{\boldsymbol{u}} \end{aligned} \tag{2.37}$$

Principle of Optimality

Suppose $\boldsymbol{u}^* \in \hat{\mathcal{C}}[t_0, t_f]^{n_x}$ it an optimal control for the problem

$$\begin{aligned} &\mathit{minimize}: \int_{t_0}^{t_f} \mathcal{L}(t, \boldsymbol{x}, \boldsymbol{u}) dt \\ &\mathit{subject\,to}: \dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}); \quad \boldsymbol{x}(t_0) = \boldsymbol{x_0} \end{aligned} \tag{2.38}$$

and suppose $\boldsymbol{x}^* \in \hat{\mathcal{C}}^1[t_0, t_f]^{n_x}$ denote the optimal corresponding response. Then, for any $t_1 \in [t_0, t_f]$, the restriction $\boldsymbol{u}^*(t)$, $t_1 \leq t \leq t_f$ is an optimal control for the problem

$$\begin{aligned} &\mathit{minimize}: \int_{t_1}^{t_f} \mathcal{L}(t, \boldsymbol{x}, \boldsymbol{u}) dt \\ &\mathit{subject\,to}: \dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}); \quad \boldsymbol{x}(t_0) = \boldsymbol{x_0} \end{aligned} \tag{2.39}$$

2.3.2 Linear Regulator Problems

An important class of optimal control problems is the one referred as linear regulator systems. In fact, it is possible to show that for linear regulator problems, the optimal control law can be found as a linear time-varying function of the state of the system. In some conditions, the control law can become time-invariant. The results here presented were shown from Kalman [54, 55] for the first time. The dynamical system is:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{2.40}$$

in this case, to simplify the notation the dependance on time has been neglected, but the state and control variable and also their coefficients $\boldsymbol{A}, \boldsymbol{B}$ can be time-varying. The performance measure is:

$$J = \frac{1}{2}\boldsymbol{x}^T(t_f)\boldsymbol{H}\boldsymbol{x}(t_f) + \frac{1}{2}\int_{t0}^{t_f}\left[\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{u}^T Ru\right]dt \tag{2.41}$$

where the final time t_f is fixed, $\boldsymbol{H}, \boldsymbol{Q}$ are real, symmetric, and positive semi-definite matrices and $\boldsymbol{R}$ is real, symmetric, positive definite matrix. The performance measure can be read as: the desire is to keep the state vector close to the origin without excessive control effort. The Hamiltonian is

$$\begin{aligned}\mathcal{H}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t) =& \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \frac{1}{2}\boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u}\\ & \boldsymbol{\lambda}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{\lambda}^T\boldsymbol{B}\boldsymbol{u}\end{aligned} \tag{2.42}$$

and necessary conditions for optimality are

$$\begin{aligned}\dot{\boldsymbol{x}}^* =& \boldsymbol{A}\boldsymbol{x}^* + \boldsymbol{B}\boldsymbol{u}^*\\ \dot{\boldsymbol{\lambda}}^* =& -\frac{\partial\mathcal{H}}{\partial\boldsymbol{x}} = -\boldsymbol{Q}\boldsymbol{x}^* - \boldsymbol{A}^T\boldsymbol{\lambda}^*\\ \boldsymbol{0} =& \frac{\partial\mathcal{H}}{\partial\boldsymbol{u}} = \boldsymbol{R}\boldsymbol{u}^* + \boldsymbol{B}^T\boldsymbol{\lambda}\end{aligned} \tag{2.43}$$

The third equation can be solved for $\boldsymbol{u}^*$, leading to:

$$\boldsymbol{u}^* = -\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{\lambda}^* \tag{2.44}$$

where the existence of $\boldsymbol{R}^{-1}$ is assured by the fact the it is positive definite. Equation (2.44) can be substituted in the first one of equations (2.43) leading to:

$$\dot{\boldsymbol{x}}^* = \boldsymbol{A}\boldsymbol{x}^* - \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{\lambda}^* \tag{2.45}$$

Hence, there are $2n$ linear homogeneous differential equations

$$\begin{bmatrix}\dot{\boldsymbol{x}}\\ ---\\ \dot{\boldsymbol{\lambda}}\end{bmatrix} = \begin{bmatrix}\boldsymbol{A} & -\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\\ -- & --------\\ -\boldsymbol{Q} & -\boldsymbol{A}^T\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{x}}\\ ---\\ \boldsymbol{\lambda}\end{bmatrix} \tag{2.46}$$

The solution to these equations has the form:

$$\begin{bmatrix}\dot{\boldsymbol{x}}^*(t_f)\\ ---\\ \dot{\boldsymbol{\lambda}}^*(t_f)\end{bmatrix} = \boldsymbol{\varphi}(t_f, t)\begin{bmatrix}\dot{\boldsymbol{x}}^*\\ ---\\ \boldsymbol{\lambda}^*\end{bmatrix} \tag{2.47}$$

where $\boldsymbol{\varphi}$ is the transition matrix of the system. Partitioning the transition matrix, we have:

$$\begin{bmatrix}\dot{\boldsymbol{x}}^*(t_f)\\ ---\\ \dot{\boldsymbol{\lambda}}^*(t_f)\end{bmatrix} = \begin{bmatrix}\boldsymbol{\varphi}_{11}(t_f, t) & \boldsymbol{\varphi}_{12}(t_f, t)\\ --- & ---\\ \boldsymbol{\varphi}_{21}(t_f, t) & \boldsymbol{\varphi}_{22}(t_f, t)\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{x}}^*\\ ---\\ \boldsymbol{\lambda}^*\end{bmatrix} \tag{2.48}$$

with $\boldsymbol{\varphi}_{11}, \boldsymbol{\varphi}_{12}, \boldsymbol{\varphi}_{21}, \boldsymbol{\varphi}_{22}$ $n \times n$ matrices. From the boundary-condition equations,

$$\boldsymbol{\lambda}^*(t_f) = \boldsymbol{H}\boldsymbol{x}^*(t_f) \tag{2.49}$$

and by substituting it in the (2.48), and by doing some easy math it can be find

$$\boldsymbol{\lambda}^*(t) = \Big[\boldsymbol{\varphi}_{22}(t_f, t) - \boldsymbol{H}\boldsymbol{\varphi}_{12}(t_f, t)\Big]^{-1}\Big[\boldsymbol{H}\boldsymbol{\varphi}_{11}(t_f, t) - \boldsymbol{\varphi}_{21}(t_f, t)\Big]\boldsymbol{x}^* \tag{2.50}$$

Kalman proved that the required inverse exists for all $t \in [t_0, t_f], so:$

$$\boldsymbol{\lambda}^*(t) = \boldsymbol{K}\boldsymbol{x}^*(t) \tag{2.51}$$

so that:

$$\begin{aligned} \boldsymbol{u}^*(t) &= -\boldsymbol{R}^{-1}(t)\boldsymbol{B}^T(t)\boldsymbol{K}(t)\boldsymbol{x}(t) \\ &= \boldsymbol{F}(t)\boldsymbol{x}(t) \end{aligned} \tag{2.52}$$

This means that the optimal control law is linear. To determine the feedback gain matrix $\boldsymbol{F}$ there are several techniques. One of these techniques shows that matrix $\boldsymbol{K}$ satisfies the matrix differential equation

$$\dot{\boldsymbol{K}} = -\boldsymbol{K}\boldsymbol{A} - \boldsymbol{A}^T\boldsymbol{K} - \boldsymbol{Q} + \boldsymbol{K}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{K} \tag{2.53}$$

with the boundary condition $\boldsymbol{K}(t_f) = \boldsymbol{H}$. This matrix differential equation is called the *Riccati equation.*

2.3.3 Numerical methods for Optimal Control Problems

In most of the cases, the dynamical system, the cost functional and the constraints are not easy to solve and they cannot be solved by 'simple' methods like the linear quadratic regulator and its derived techniques. In all the cases in which simple techniques are not available, numerical methods must be used to solve the optimal control problem (OCP). With the development of high speed computers in last decades it has become possible to solve also complex problems in a reasonable amount of time. The field of numerical methods for OCPs is really wide and it is also impossible to cover it all, but the methods can be divided onto two main categories [56, 57]:

- Direct solution methods
- Indirect solution methods

The methods in the first class try to find a minimum of the objective function by constructing a sequence of points converging to that minimum. The second class is also the one referred as variational methods in the literature, and this is because their purpose is to find the solution of the equation, resulting from the application of the necessary optimality conditions $\delta J(\boldsymbol{u}) = 0$. Another way to put it, is that direct methods require only a comparison between the value of the functional with respect to the previous iteration. Indirect methods seek the solution of the necessary condition. Direct methods solve the OCP first discretizing the problem and then optimizing it. More precisely, direct approch transform the infinite horizon problem into a finite dimensional one. Then, the nonlinear programming problem resulted is proved to be an equivalent representation of the original one. Lastly, a nonlinear programming solver is used to find the optimal trajectory. The indirect methods apply the Pontryagin's Maximum Principle together with the transversality condition and obtain the necessary optimality conditions. If, as almost always happens, the initial values for the state variable are defined, the result is the complete description of the system. One of the biggest problem in indirect methods are referred to the nonlinearities in the dynamical system, since they can produce difficulties in the application of the Pontryagin Maximum Principle. Therefore, one way to overcome this problem is to linearize the dynamic system around a convenient point and then to study the simplified version of the dynamic system. As always, even though there are many different linearization techniques, they can be fallacious.

Principal Indirect Method Techniques

In these methods, the calculus of variation is used to determine the first-order optimality condition of the optimal control problem, which leads to the first-order necessary conditions discussed in the previous section. The three most common indirect methods are:

- shooting method;
- multiple-shooting method;
- collocation method;

Indirect shooting method
The first one is the most basic one. Since variational approach leads to ad unknown boundary condition at one end of the interval, a guess of this unknown condition is made. The guess is used together with the initial condition (assumed known). The Hamiltonian system is integrated to the other end (forward or backward). Once the terminal end is reached, the terminal condition obtained from the numerical integration is compared with the known value. If the difference between the obtained and the known values is lower than a chosen threshold, the method ends. If the difference is greater than the specified threshold, an adjustment on the initial guess is made and the method is launched again.
Indirect multiple shooting method
The shooting method described above is really appealing due its simplicity, but it usually presents wide numerical difficulties due to ill-conditioning of the Hamiltonian dynamics. This is because an even small error on the initial guess can result in a huge amplification of the error when the dynamics is integrated in time, regardless the integration is performed forward or backward. To overcome some of the numerical issues of shooting method, the multiple shooting method has been developed. The time interval is divided into $M + 1$ subintervals and the shooting method is then applied over each subinterval. To enforce continuity, following conditions are imposed

for each subinterval:

$$\begin{aligned}\boldsymbol{x}(t_i^-) &= \boldsymbol{x}(t_i^+)\\ \boldsymbol{\lambda}(t_i^-) &= \boldsymbol{\lambda}(t_i^+)\end{aligned} \tag{2.54}$$

Despite the need of introducing more equations, the multiple shooting method is an improvement with respect to the 'simple' shooting method, since the numerical error during the integration is smaller, since the time interval in which the integration is performed is smaller. Nevertheless, even this method can present problems if the initial guess for the costate is not sufficiently good.

Indirect collocation method

In this method the state and costate are parameterized using piecewise polynomials. This procedure leads to a root-finding problem, where the unknowns are the coefficients of the piecewise polynomial. Therefore, this method is than solved using an *ad-hoc* chosen root-finding technique.

Principal Direct Method Techniques

Direct methods are very different from indirect ones. The state and/or the control of the original optimal control problem is approximated in some manner. These techniques are different if only the control is approximated (control parameterization methods) or if both the control and the state are approximated (state and control parameterization methods). In both cases, the optimal control problem becomes a nonlinear optimization problem, also referred in literature as nonlinear programming problem (NLP).

Direct shooting method

Once again, the most basic method is the direct shooting method. It is a control parameterization technique in which the control is assumed in the form:

$$\boldsymbol{u}(t) \approx \sum_{i=1}^{m} \boldsymbol{a}_i \psi_i(t) \tag{2.55}$$

where $\psi_i(t)$ are known functions and $\boldsymbol{a}_i$ are the parameters to be determined from the optimization. The dynamics are then satisfied by integrating the differential equation, while the cost function is determined using a quadratic approximation. The NLP minimizes the cost subject to path and interior-point constraints.

Direct multiple shooting method

In a similar manner to the indirect method, the time interval is divided into $M+1$ subintervals. The direct method described above is then used over each subintervals. To enforce the continuity of the solution, it is imposed that:

$$\boldsymbol{x}(t_i^-) = \boldsymbol{x}(t_i^+) \tag{2.56}$$

As in the indirect case, the multiple-shooting technique increases the size of the problem. Despite the increasing in the size, the multiple-shooting method is an improvement with respect to the 'simple' shooting one, because the integration is performed over significantly smaller time intervals.

Direct collocation method

Perhaps this one is the most powerful of the discussed methods. The direct collocation method is based on a parameterization of both the control and the state variables. There are two different forms of collocations, which are local and global collocation. A local collocation methods follow a similar scheme with respect to the multiple shooting direct methods, with the division of the integral in smaller subintervals and the continuity constrained applied on each subinterval(here usually referred as *compatibility*). The discretization is then performed, using for instance Runge-Kutta methods. The remarkable advantage of this strategy is that this method leads to a large sparse NLP, i.e. the NLP has thousands (even tens of thousands) of variable and a similar number of constraints, but many of the derivatives of the constraint Jacobian are zero due the sparsity of the problem. This leads the solution of the NLP to be rather 'easy' to solve with a proper machine solver.

2.4 Motivation for a new study

The new study here introduced presents the Feedback Local Optimality Principle, or FLOP. The method moves from the Optimal Control Problems and uses the power of the variational calculus to provide an innovative feedback control, based on the variational principles [40]. The classical approach to Optimal Control Problems provide the minimization of a performance measure, i.e. the functional J. Through the minimization, the optimal trajectories both for control, state and adjoint vectors are obtained, u^*, x^*, and λ^* respectively. However, these techniques present some drawbacks for engineering applications:
- the optimal trajectory can be very difficult (even impossible) to find analytically
- if an optimal trajectory exists, the resulting control law is a feed-forward law.

Having a feed-forward law as control law gives problems in terms of reliability and robustness of the solution, since feed-forward law cannot take account of disturbances during the process or errors in the mathematical model. In fact, feed-forward laws give perfect responses when the system's model is not affected by any errror, the state of the system is perfectly known and all the external forces are known in advance. Unfortunately, the engineering world very often does not present any of the requirements for the success of the feed-forward control law, which can be used in some test-case or in some very well-posed problems. In general models, the controlled process presents some degrees of approximation since the dynamics is roughly represented by the estimated differential equations, the external forces are generally unknown and some disturbances are generally present. With the increasing of the difficulty of the problem, the feed-forward law is not enough for the solution of the problem, since the presence of disturbances and of the complexity of the dynamical system. Hence, complex optimal control problems, i.e. those which arise from applications in modern engineering in fields as aeronautics, astronautics, robotics, mechatronics, can today be solved by advanced numerical methods. One of the most used techniques is the multiple shooting method, which has been tested in difficult problems, such as [58–62]

However, multiple shooting techniques is generally referred as very complex to use, since a deep knowledge of the physical problem and of calculus of variation is necessary to assign a reasonable guess for the initial condition of the iteration process. In the indirect methods there is also to deal with the adjoint variables, which is again

affected by the lack of information about it. Moreover, the convergence method used in indirect methods (for instance the Newton method) has a relatively small domain of convergence. These difficulties are typical for indirect methods, with the additional complication of the sequence of boundary problems resulting from the main principle of the multiple shooting technique, i.e. the division of the original integral into an high number of subintervals. With a completely different approach, the OCP can be transformed into a nonlinear programming problem using a direct approach, using the parameterization of the control variables. Different choices can be made: one possible idea is to perform an explicit integration of the equation of motion, choosing the control variable from a finite dimensional space. The explicit integration, though, can be avoided if the state variable is either parameterized or discretized. In this last case, the equations of motion are then satisfied only pointwise by using the so-called collocation conditions. The advantages of the indirect multiple shooting technique reside in the accuracy and the possibility to verify many necessary conditions. In recent years methods for the developing of indirect methods with parallel computers were developed, so that the computational time may decreases. The great advantage in direct methods is in the avoidance of the explicit integration, which leads to a very efficient process in terms of computational time.
In this work the innovative FLOP control is presented. The objective of the study was to find a control law which may have the following strengths:

- the control law must be in feedback;
- the control law must require low computational cost;
- the control law must be applicable in a wide range of applications.

The necessity of the feedback control law relies behind the desire of having a control which can be applied in real engineering problems. In fact, in real problems the model of the process is affected by some errors and by some simplifications, and some disturbance is always present. Moreover, during the dynamical evolution it is possible that some unexpected conditions (i.e. an obstacle on the path) require some sudden changes (i.e. a quick steering). On the one hand, the initial will was to use the power of the calculus of variation, which can lead to a solid mathematical basis for the innovative control and it gives, through the construction of the functional, a high flexibility to the number of possible applications. On the other hand, with the classical variational approach and the resulting feed-forward control law, it is impossible to account for errors in the model, for external noises, and for unexptected conditions as well. Therefore, the strategy with the FLOP control is to use the calculus of variation, but also to change something in the original formulation, in order to provide a feedback control law. The problem of the computational cost is related to the will of having a control which can be applied in real time applications, such as the autonomous driving, swarm robotics and so on.
The main intuition behind the FLOP is to take advantage of both the direct and the indirect methods for solving OCPs:
first, the original functional is divided into N sub-intervals, just as in the multiple shooting indirect techniques . One of the principal advantages of the division of the problem into an high number of sub-problems is the decreasing of the integration

error, since the time interval of each integral is much lower than the one in the original problem. The disadvantage, of course, is referred to the high number of guesses that this strategy generally needs, along with the continuity conditions required for each interval.
To overcome the difficulties of the indirect method strategies, the idea is to use a parameterization of the adjoint variables, which permits to reduce dramatically the number of shooting necessary in the solution technique. To perform the parameterization some conditions have to be satisfied:
- the dynamical system must be in the form: $\dot{\boldsymbol{x}} = \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{B}\boldsymbol{u}$
- the cost function must be in the form: $E(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u} + g(\boldsymbol{x})$
- a local optimum has to be accepted instead of the global optimum of the classical theories

In these conditions, the resulting technique is an hybrid technique which tries to take the best from the calculus of variations and both the direct and the indirect methods. The FLOP control succeeds, under the above written conditions, in performing a feedback control law which can be used in a wide range of real engineering applications.
In next chapter will be presented the innovative feedback control named Feedback Local Optimality Principle. First, an intuitive presentation of the method will be introduced. Then, a more precise mathematical presentation will be discussed and the FLOP will be derived both in the one-dimensional case and then in the multi-dimensional case.
Then, some of the possible applications of the FLOP method are treated: prototype problems, the obstacle avoidance strategy for an autonomous vehicle, the rescue of a breakdown boat from a swarm of marine drones. In all these applications, the possibility given by the FLOP method to produce a feedback control law for a nonlinear dynamical system, using highly nonlinear penalty function is shown.

Chapter 3

A new feedback approach for optimal control

In this chapter an innovative feedback approach, named Feedback Local Optimality Principle (FLOP) for optimal control problem is presented. The classical theories such as the Pontryagin's and Bellman's ones present some drawbacks from an engineering point of view, since they succeed in solve the OCP with its global optimum in terms of state and control trajectories, but at the cost of a feed-forward control law. A feed-forward law is not able to account any errors in the model or any external disturbances, therefore they are not widely used in engineering applications. OCPs are also solved using numerical solutions, such as the direct and indirect methods discussed in previous sections. Even these strategies are not very efficient, since in the direct methods a strong parameterization of the variables is necessary, and an error must be accounted, while the indirect methods often requires a deep knowledge of variational calculus and multiple guesses for some variables, leading to high computational costs. The idea of the FLOP approach is to use the power of the calculus of variation as in classical theories, and some advantages both from direct and the indirect methods, to provide a feedback control law for a certain class of problems. In next section, first the class of possible problems in which FLOP approach can be used is individuated, together with the problem statement. Then, an intuitive presentation of the approach is given, following the main idea behind the approach. Finally, the rigorous formulation both for the monodimensional and the multidimensional approach is developed.

Problem statement

In section 2.3, variational problems were presented. In general, they are based on three elements: the cost function $E(x, u)$, the dynamical system $\dot{x} = f(x, u)$, and the optimality principle. In general, OCT is based on the minimization or maximization of a performance index $\bar{J}$, defined as:

$$\begin{gathered}\min \bar{J} = \int_0^T E(x,u)dt \\ \text{subject to} \\ \dot{x} = f(x,u) \\ x(0) = x_0\end{gathered} \tag{3.1}$$

in equation (3.1), x, u are the state and control variables respectively. The controlled process is $\dot{x} = f(x,u)$, and $x(0) = x_0$ its associated initial condition. The dynamical evolution of the system represents a constraint between x and u. In the Euler-Lagrange formulation, the constraint is included through the Euler-Lagrange multiplier λ, and $\bar{J}$ can be written as:

$$\begin{gathered}\min \bar{J} = \int_0^T \Big[E(x,u) + \lambda(\dot{x} - f(x,u))\Big]dt \\ \text{subject to} \\ x(0) = x_0\end{gathered} \tag{3.2}$$

The above integral can be written for simplicity as:

$$\min \bar{J} = \int_0^T \Big[E(x,t) + \lambda(\dot{x} - f(x,u))\Big]dt = \int_0^T \mathscr{L}(\dot{x}, x, u, \lambda)dt \tag{3.3}$$

The problem in equation(3.3) is solved when the optimal trajectory x^* associated to the optimal control u^* is found. The general solution can be represented in an abstract form by $v = (x,u)$, and therefore $v^* = (x^*, u^*)$. Given the minimization problem in (3.1), $\bar{J}(x,u) \geq \bar{J}^*((x^*, u^*)) \; \forall \; v \;\neq\; v^*$. One of the possible methods to solve the problem in (3.3) is the classical variational approach, which leads to the Pontryagin's equations:

$$\begin{cases}\nabla_x E - \lambda \nabla_x f - \dot{\lambda} = 0 \\ \nabla_u E - \lambda \nabla_u f = 0 \\ \dot{x} = f(x,u) \\ x(0) = x_0 \\ \lambda(T) = 0\end{cases} \tag{3.4}$$

where the fourth equation represents the so-called transversality condition. The solution can be represented as:

$$v^* = (x^*, u^*) = V(x_0, T, t) \tag{3.5}$$

where V describe the structure of the solution. In fact, once the cost function E, the dynamical evolution of the system f, are assigned in (3.4), the solution depends only on the two parameters x_0 and T. Hence, the integral can be written for simplicity as:

$$\begin{gathered}\min \bar{J} = \int_0^T \mathscr{L}(v)dt \\ \text{subject to} \\ x(0) = x_0\end{gathered} \tag{3.6}$$

The Pontryagin's solution (3.5) has some weaknesses from an engineering point of view, as previously discussed. To overcome some of these weaknesses, the idea is to solve a similar problem respect to the Pontryagin's one but, by relaxing the global optimality condition into a local optimality condition, obtaining a more suitable control law for engineering purposes.

The Feedback Local Optimality Principle (FLOP) is an innovative approach for optimal control problems, which can be used for the affine class of dynamical systems in the form:

$$f(x,u) = \phi(x) + bu \tag{3.7}$$

FLOP is derived starting from the classical variational approach, but by changing the optimality principle and accepting a local minimum instead of the global one, succeeds in producing a new feedback control law for dynamical systems in the form of equation (3.7), if the cost function can be written in the form:

$$E(x,u) = g(x) + ru^2 \tag{3.8}$$

Therefore, the general problem for the FLOP method is:

$$\min \bar{J} = \int_0^T g(x) + ru^2 + \lambda(\dot{x} - (\phi(x) + bu))dt \tag{3.9}$$

The change from seeking a local optimum instead of the global one is performed using a parameterization of the adjoint variable λ, but it relies behind the idea of dividing the original integral of the problem into N sub-intervals and by optimizing each one of them. This procedure, later explained in details, succeeds in providing a feedback control law for the (3.9). The main advantages of the FLOP method are:

- **feedback control law**: having a closed-loop control law permits the usage of the FLOP method for real engineering applications, since both errors in the model and external unknown disturbances can be account.
- **flexibility**: since the general formulation requires the dynamical system to be linear on the control variable and the cost function to be quadratic in the control variable, but there are no requirements on the state variable, the FLOP technique can be applied on a wide range of applications.
- **computational costs**: the FLOP strategy permits to have a control law with low computational costs, so that it can also be applied in real time applications.

The FLOP formulation present some drawbacks, the first one is related to the acceptance of the local optimum instead of the global one, so the resulting control and state trajectories are not the optimal ones. The second disadvantage regards the necessary parameterization of the adjoint variable λ, which leads to a parameterization for the control variable u. Despite these drawbacks, the FLOP method proved to be very useful in many different applications. In factmmh, in the next chapter FLOP approach will be tested against other well-known techniques in prototype problems, revealing very promising results, and finally it will be used to control an unmanned ground vehicle in a crash avoidance and a swarm of boats for the rescue of a breakdown member of the swarm.

3.1 An intuitive presentation

In this paragraph, an intuitive presentation of the Feedback Local Optimality Principle is presented. As in the multiple shooting techniques, the functional $\bar{J}$ in formulation (3.3) is divided into N integrals with fixed time-interval $\Delta\tau$, so that $\Delta\tau_i = \Delta\tau = T/N$. By rewriting equation (3.3) with this new formulation, it follows:

$$\bar{J} = \int_{LB_1}^{UB_1} \mathscr{L}(v_1)dt + \cdots + \int_{LB_i}^{UB_i} \mathscr{L}(v_i)dt + \cdots + \int_{LB_N}^{UB_N} \mathscr{L}(v_N)dt \tag{3.10}$$

where $LB_i = (i-1)\Delta\tau$ and $UB_i = (i)\Delta\tau$. By minimizing each integral in equation (3.10), a new minimum $\bar{J'}^*$ is found and in general it results

$$\bar{J'}^* \geq \bar{J}^* \tag{3.11}$$

The inequality in (3.11) is reasonable, since a sequence of minimization does not converge in the global minimization previously performed (with the exception of simple test case examples). The solution of problem (3.10) can be written accordingly to (3.4) and using the proper transversality and continuity condition, $\lambda_i = \lambda|_{UB_i} = 0$ and $x_{LB_i} = x_{UB_i}$ respectively, it becomes:

$$\begin{cases} \nabla_{x_i} E - \lambda_i \nabla_{x_i} f - \dot{\lambda_i} = 0 \\ \nabla_{u_i} E - \lambda_i \nabla_{u_i} f = 0 \\ \dot{x_i} = f(x_i, u_i, t) & \forall\, t \,\in\, [LB_i, UB_i] \\ x_{LB_i} = x_{UB_i} \\ \lambda(UB_i) = 0 \end{cases} \tag{3.12}$$

It is important to note that the problem in (3.12) is not different from the original one, in terms of complexity. In fact, the only difference with the original problem is the different time interval for the integration. Nevertheless, since the time interval is much smaller than in the original problem, the differential equations in the above system can be linearized at the first order. In particular, as discretization step here the choice is to use $\Delta\tau$, which leads to:

$$\begin{cases} \nabla_x E|_{LB_i} - (\lambda \nabla_x f)|_{LB_i} - \frac{\lambda_{UB_i} - \lambda_{LB_i}}{\Delta\tau} = 0 \\ \nabla_u E|_{LB_i} - (\lambda \nabla_u f)|_{LB_i} = 0 & \forall\, i \,\in\, [1, N] \\ \frac{x_{UB_i} - x_{LB_i}}{\Delta\tau} = f(x_{LB_i}, u_{LB_i}, (i-1)\Delta\tau) \end{cases} \tag{3.13}$$

From system (3.13) it is important to consider that x_{LB_i} is known for reason explained in a moment, while $\lambda_{UB_i} = 0$ for each i between $[1, N]$, so that each integral has to be minimized. The solution can be summarized as follows: starting from $i = 1$, continuity and transversality conditions are set so that: $x_{LB_1} = x_0$, $\lambda_{UB_1} = 0$. Given these two conditions, equations in (3.13) become a system of three equations in three unknowns. For some cases of $f(x, u, t)$ that will be later study in deep, equation (3.13) can be analytically solved. If a solution for (3.13) exists for $i = 1$, new continuity and transversality conditions are set so that: $x_{LB_2} = x_{UB_1}$, $\lambda_{UB_2} = 0$, and system in (3.13) can be solved iteratively for $i = 2,\ 3, \ldots, N$. In Fig. 3.1, the diagram of the FLOP optimization process is depicted for each interval, represented as a rectangular box. It is important to denote that the choice of $\Delta\tau$ plays an important role in the presented technique. In fact, its choice affects the degree of accuracy of the obtained solution. The choice of the best possible value for $\Delta\tau$ is part of the rigorous approach presented in the next section.

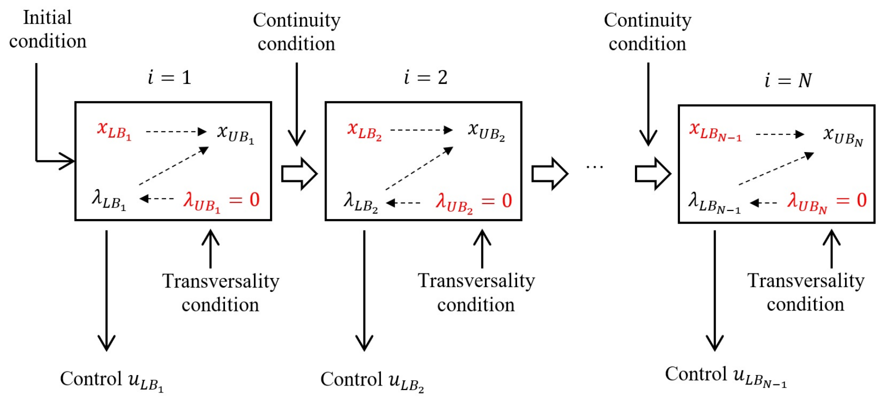

Figure 3.1. FLOP scheme

The intuitive presentation of the FLOP strategy presents the division of the original integral into an elevated number of integrals as in the indirect multiple shooting methods, and similarly it uses the variational approach for each integral. As it was pointed out, the new problem is in any way easiest than the original, but using the fact that the time interval for each integral is little, a first order discretization is performed on the remaining differential equations. By applying the local transversality condition (i.e. by seeking a local minimum for each integral) it is possible to obtain a control law for each integral. One of the advantages of this procedure relies in the fact that the derivatives in the system (3.13) does not require to be evaluated for each $i \in [1, N]$, since they are always the same. The only necessary procedure is to evaluate the continuity condition $x_{UB_i} = x_{LB_{i-1}}$, and to choose the integer number N in which the original integral is divided. With respect to the classical multiple shooting methods, the great advantage here is that the FLOP procedure leads to a single shooting technique, since the only parameter to be tuned remains N, and $\Delta\tau$ as a direct consequence. Nevertheless, since the imposition of the transversality condition for each integral, each one of them is minimized and the original functional is evaluated as a sum of minimized functionals. From a computational point of view, the FLOP approach does not require high performances, since all the derivatives can be evaluated offline before the process starts.

At this point, it is important to denote that the given control techniques is a discrete technique, which can present a number of liabilities in engineering applications, such as:

- no control from one time-step to another
- discretization of the dynamical process

To avoid both these two main disadvantages, the intuitive procedure is extended in a different solution technique, which starts from the intuitive procedure: Since for each i the differential equation of the adjoint variable is written as:

$$\dot{\lambda}_i = \frac{\lambda_{UB_i} - \lambda_{LB_i}}{\Delta\tau} \tag{3.14}$$

but the transversality condition $\lambda_{UB_i} = 0$ is applied, this implies that for each i it can be written

$$\dot{\lambda}_i \approx -\frac{\lambda_{LB_i}}{\Delta\tau} \tag{3.15}$$

This lead to test a continuous counterpart for the parameterization of the adjoint variable, so that:

$$\dot{\lambda} = -\frac{\lambda}{\Delta\tau} = G\lambda \tag{3.16}$$

Using this new parameterization for the lagrangian multiplier λ, the set of variations in (3.4) can be rewritten as:

$$\begin{cases} \nabla_x E - \lambda \nabla_x f - \dot{\lambda} = 0 \\ \nabla_u E - \lambda \nabla_u f = 0 \\ \dot{x} = f(x,u) \\ \dot{\lambda} = G\lambda \end{cases} \qquad \forall\, t \in [0,T] \tag{3.17}$$

where as always $x(0) = x_0$, with x_0 assumed to be known. This formulation can be used if E, f are in the forms described above, which here are rewritten for convenience for the reader:

$$\begin{aligned} E(x,u) &= q(x) + ru^2 \\ f(x,u) &= \phi(x) + bu \end{aligned}$$

The above conditions permit to the FLOP method to intervene in different dynamical systems and in different scenarios, giving a great flexibility on the different problems which can be treated with this innovative strategy. The control variable can be easily derived from the previous formulation. In fact, with some easy calculations, the FLOP control in this monodimensional case is given by the sequent law:

$$u(x) = \frac{b}{r}(G + \nabla_x \phi)^{-1} \nabla_x g \tag{3.18}$$

The continuous counterpart of the set of equations written in the system in (3.17), permit to overcome the problems of the discrete approach, which is here given in order to guide the reader in the idea which led to the FLOP formulation. The FLOP formulation is an augmented form of the classical Pontryagin's formulation for Optimal Control Problems. By performing the parameterization of the adjoint variable, the FLOP strategy obtains a feedback control law in the form given in equation (3.18). The FLOP formulation, here presented in the monodimensional case, can be extended in the multidimensional case. Before the rigorous approach which will be given in next sections, the FLOP formulation in the multidimensional case is here given. Starting from the integral:

$$\min \bar{J} = \int_0^T g(\boldsymbol{x}) + \boldsymbol{u}^T \boldsymbol{R}\boldsymbol{u} + \boldsymbol{\lambda}^T(\dot{\boldsymbol{x}} - (\boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{B}\boldsymbol{u}))dt \tag{3.19}$$

the augmented form of the Potryagin's formulation becomes:

$$\begin{cases} \nabla_x g - \nabla_{\boldsymbol{x}}\boldsymbol{\phi}^T\boldsymbol{\lambda} - \dot{\boldsymbol{\lambda}} = \boldsymbol{0} \\ \boldsymbol{R}^T\boldsymbol{u} - \boldsymbol{B}^T\boldsymbol{\lambda} = \boldsymbol{0} \\ \dot{\boldsymbol{x}} = \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{B}\boldsymbol{u} \\ \dot{\boldsymbol{\lambda}} = \boldsymbol{G}\boldsymbol{\lambda} \end{cases} \quad \forall\, t \in [0, T] \tag{3.20}$$

where $\boldsymbol{G} = -\frac{\boldsymbol{I}}{\Delta_T}$. The multidimensional feedback control law becomes:

$$\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{R}^{-T}\boldsymbol{B}^T(\boldsymbol{G} + \nabla_{\boldsymbol{x}}\boldsymbol{\phi})^{-1}\nabla_{\boldsymbol{x}} g \tag{3.21}$$

3.2 A rigorous variational approach for the feedback control: monodimensional case

In the previous section, the intuition which led to the formulation of the Feedback Local Optimality Principle is given. The choice to relax the global optimality condition of classical theory into a sequence of local optimality condition permits to express the control law u at each time-step as a function of E and f, both evaluated at the previous time-step, resulting in a feedback control law. As it has been said, FLOP method does not work with any dynamical systems and with any cost function. In fact, since the requirement is that three equations in (3.13) can be solved with three specified unknowns, the dynamical system needs to be in the form $f(x, u) = \phi(x) + bu$ and cost function $E(x, u) = q(x) + bu^2$. Starting from the intuitive presentation of the FLOP method, and with f and E assigned, a more rigorous formulation for the Feedback Local Optimality Principle is developed, starting from the one-dimensional case. The problem statement is here slightly changed and two new constraints are added, so that:

$$J = \int_0^T \Big[q(x) + \frac{1}{2}r(u - u_T)^2 + \lambda(\dot{x} - \phi(x) - bu) + \lambda\dot{g} + \frac{1}{2}\lambda g^2\Big]dt \tag{3.22}$$

Performing the variations on (3.22), the augmented form presented is re-obtained:

$$\begin{cases} q_x - \dot{\lambda} - \lambda\phi_x = 0 \\ r(u - u_T) - \lambda b = 0 \\ \dot{x} - \phi - bu + \dot{g} + \frac{1}{2}g^2 = 0 \\ \dot{\lambda} - g\lambda = 0 \end{cases} \tag{3.23}$$

with:

$$x(0) = x_0$$
$$\lambda(T) = 0$$
$$g(0) = g_0$$

where $\phi_x = \nabla_x\phi(x)$ and $q_x = \nabla_x q(x)$. The second equation in (3.23) can be used to explicit the control variable u, and the system becomes:

$$\begin{cases} \dot{\lambda} = q_x - \lambda\phi_x \\ \dot{x} + \dot{g} = \phi + b(\frac{b}{r}\lambda + u_T) - \frac{1}{2}g^2 \\ \dot{\lambda} = g\lambda \end{cases} \tag{3.24}$$

where it is used $u = \frac{b}{r}\lambda + u_T$. The system in (3.24), can not be solved, since does not satisfies the Cauchy criterion, which states that a system in the form $\boldsymbol{A}\dot{\boldsymbol{\psi}} = \boldsymbol{\rho}(\boldsymbol{\psi})$ can be solved if and only if $\boldsymbol{A}$ can be inverted. In the case in (3.24),

$$\boldsymbol{A}\dot{\boldsymbol{\psi}} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{g} \\ \dot{\lambda} \end{bmatrix} \tag{3.25}$$

The first and the third equations in the system (3.24) are here combined, so that:

$$\lambda = \frac{q_x}{g + \phi_x} \tag{3.26}$$

from which the control variable u is re-evaluated as:

$$u = \frac{b}{r}\frac{q_x}{g + \phi_x} + u_T \tag{3.27}$$

The system (3.24) is here rewritten, using equation (3.26) and (3.27) in the second and the third equation, and a new system is obtained:

$$\begin{cases} \dot{x} + \dot{g} = \phi + \frac{b^2}{r}(\frac{q_x}{g+\phi_x}) + bu_T - \frac{1}{2}g^2 \\ q_{xx}\frac{\dot{x}}{g+\phi_x} - \frac{q_x}{(g+\phi_x)^2}(\dot{g} + \phi_{xx}\dot{x}) = g\frac{q_x}{(g+\phi x)} \end{cases} \tag{3.28}$$

System (3.28) can be written in the Cauchy form:

$$\begin{cases} \dot{x} + \dot{g} = \phi + \frac{b^2}{r}(\frac{q_x}{g+\phi_x}) + bu_T - \frac{1}{2}g^2 \\ \dot{x}\Big[q_{xx}(g + \phi_x) - q_x\phi_{xx}\Big] - q_x\dot{g} = gq_x(g + \phi_x) \end{cases} \tag{3.29}$$

which means that:

$$\tilde{\boldsymbol{A}}\dot{\boldsymbol{\psi}} = \begin{bmatrix} 1 & 1 \\ q_{xx}(g + \phi_x) - q_x\phi_{xx} & q_x \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{g} \end{bmatrix} \tag{3.30}$$

The determinant of $\tilde{\boldsymbol{A}}$ needs to be non-zero, therefore:

$$q_x \neq q_{xx}(g + \phi_x) - q_x\phi_{xx} \tag{3.31}$$

and, to assure that, the following hypotesis is made:

$$\left|\frac{q_x x}{q_x}(g + \phi_x)\right| >> \left|\phi_{xx}\right| \tag{3.32}$$

Here, another strong hypotesis is made, according with the intuitive presentation of the previous section:

$$g(t) = cost = g_0 \tag{3.33}$$

With this second assumption, equation (3.29) is reworked in:

$$\begin{cases} \dot{x} = \phi + \frac{b^2}{r}(\frac{q_x}{g_0+\phi_x}) + bu_T - \frac{1}{2}g_0^2 \\ \dot{x}\Big[q_{xx}(g_0+\phi_x) - q_x\phi_{xx}\Big] = g_0 q_x(g_0+\phi_x) \end{cases} \tag{3.34}$$

Two differential equations in $\dot{x}$ need to be solved, and as in the previous case in equation (3.24), a substitution of the second one in the first one is done:

$$\left(\frac{q_{xx}}{q_x}(g_0+\phi_x) - q_x\phi_{xx}\right)\left(\phi + \frac{b^2}{r}(\frac{q_x}{g_0+\phi_x}) + bu_T - \frac{1}{2}g^2\right) = g_0\Big(g_0+\phi_x\Big) \tag{3.35}$$

and by using the hypothesis in (3.32), it is obtained:

$$q_{xx}\left(\phi + \frac{b^2}{r}(\frac{q_x}{g_0+\phi_x}) + bu_T - \frac{1}{2}g_0^2\right) = g_0 q_x \tag{3.36}$$

In general, it is always possible to write the general nonlinear and nonquadratic penalty function $q(x)$ with the Taylor expansion:

$$\begin{gathered} q(x) \approx q(x_T) + q_x(x_T)(x-x_T) + \frac{1}{2}q_{xx}(x_T)(x-x_T)^2 \\ q_x(x) \approx q_x(x_T) + q_{xx}(x_T)(x-x_T) \\ q_{xx}(x) \approx q_{xx}(x_T) \end{gathered} \tag{3.37}$$

and hence, using the Taylor expansion, (3.36) becomes:

$$\phi + \frac{b^2}{r}(\frac{q_x}{g_0+\phi_x}) + bu_T - \frac{1}{2}g_0^2 = g_0\frac{q_x(x_T)}{q_{xx}(x_T)} + g_0(x-x_T) \tag{3.38}$$

Since the idea is to have the chance to compare the right and the left member of the equation (3.38), the right one is here expanded with the Taylor series:

$$\begin{gathered} \phi + \frac{b^2}{r}(\frac{q_x}{g_0+\phi_x}) + bu_T - \frac{1}{2}g_0^2 \approx \\ \approx \phi(x_T) + \frac{b^2}{r}(\frac{q_x(x_T)}{g_0+\phi_x(x_T)}) + bu_T - \frac{1}{2}g_0^2 + \\ +\left(\phi_x(x_T) + \frac{b^2}{r}\frac{q_{xx}(x_T)(g_0+\phi_x(x_T)) - q_x(x_T)\phi_{xx}(x_T)}{(g_0+\phi_x(x_T))^2}\right)(x-x_T) \end{gathered} \tag{3.39}$$

By using the hypotesis (3.32), the expression above becomes:

$$\begin{gathered} \phi + \frac{b^2}{r}\frac{q_x}{g_0+\phi_x} + bu_T - \frac{1}{2}g_0^2 \approx \\ \phi(x_T) + \frac{b^2}{r}\frac{q_x(x_T)}{g_0+\phi_x(x_T)} + bu_T - \frac{1}{2}g_0^2 + \\ +\left(\phi_x(x_T) + \frac{b^2}{r}\frac{q_{xx}(x_T)}{g_0+\phi_x(x_T)}\right) \end{gathered} \tag{3.40}$$

By comparing the terms with the state variable dependance and the ones without it, two equations for the two unknowns are following:

$$\begin{cases} g_0 = \phi_x(x_T) + \frac{b^2}{r}\left(\frac{q_{xx}(x_T)}{g_0+\phi_x(x_T)}\right) \\ bu_T = -\phi x_T + \frac{1}{2}g_0^2 + g_0\frac{q_x(x_T)}{q_{xx}(x_T)} - \frac{b^2}{r}\frac{q_x(x_T)}{g_0+\phi_x(x_T)} \end{cases} \tag{3.41}$$

from which the two unknowns u_T and g_0 can be obtain so that:

$$\begin{aligned} g_0 &= -\sqrt{\phi_x^2(x_T) + \frac{b^2}{r}q_{xx}(x_T)} \\ u_T &= -\frac{\phi(x_T)}{b} + \frac{g_0^2}{2b} + \frac{g_0}{b}\frac{q_x(x_T)}{q_{xx}(x_T)} - \frac{b}{r}\frac{q_x(x_T)}{g_0\phi_x(x_T)} \end{aligned} \tag{3.42}$$

and the final formulation of the FLOP control in the one-dimensional case is:

$$\begin{aligned} \dot{x} &= \phi + bu \\ u &= \frac{b}{r}\frac{q_x}{g_0+\phi_x} + u_T \\ u_T &= -\frac{\phi(x_T)}{b} + \frac{g_0^2}{2b} + \frac{g_0}{b}\frac{q_x(x_T)}{q_{xx}(x_T)} - \frac{b}{r}\frac{q_x(x_T)}{g_0\phi_x(x_T)} \\ &\text{with} \\ x(0) &= x_0 \\ g_0 &= -\sqrt{\phi_x^2(x_T) + \frac{b^2}{r}q_{xx}(x_T)} \end{aligned} \tag{3.43}$$

With respect to the intuitive presentation, this rigorous one permits to identify a value for the variable g_0, while in the intuitive presentation the value of $\Delta\tau$ remained a tuning parameter. Nevertheless, it is not assured that the founded value of g_0 will be the optimal one, but the rigorous approach permits to have this starting value for the parameter to be tuned. In this section, the mathematical development of the FLOP algorithm in the monodimensional case was presented. Starting from an augmented functional with respect to the classical one, it succeeds in giving a feedback control law for a dynamical system of the form $\dot{x} = \phi(x) + bu$ if the cost function is in the form $E(x,u) = q(x) + ru^2$ and the function $q(x)$ is differentiable in the state.

3.3 A rigorous variational approach for the feedback control: multi-dimensional case

In this section, the Feedback Local Optimality Principle is extended in the multi-dimensional case. On the one hand, calculations are very similar to the one-dimensional case and the ideas behind them remains the same. On the other hand, some adjustments are needed. The statement of the problem is the same: a functional J has to be minimized and it is here written as:

$$J = \int_0^T \left[q(\boldsymbol{x}) + \frac{1}{2}(\boldsymbol{u}-\boldsymbol{u_T})^T \boldsymbol{R}(\boldsymbol{u}-\boldsymbol{u_{T})+\lambda^T(\dot{x}-\phi-Bu)+\lambda^T\dot{g}+\frac{1}{2}g^T\Lambda g}\right] dt \tag{3.44}$$

where $\boldsymbol{\Lambda} = diag(\lambda)$. The variations are performed to the functional in (3.44), and it is obtained:

$$\begin{cases} q_{\boldsymbol{x}} - \boldsymbol{\phi_x}^T \boldsymbol{\lambda} = \dot{\boldsymbol{\lambda}} & \text{with} \\ \boldsymbol{R}^T(\boldsymbol{u} - \boldsymbol{u_T}) - \boldsymbol{B}^T \boldsymbol{\lambda} = \boldsymbol{0} & \boldsymbol{x(0)} = \boldsymbol{x_0} \\ \dot{\boldsymbol{x}} = \boldsymbol{\phi} + \boldsymbol{Bu} - \dot{\boldsymbol{g}} - \frac{1}{2}\boldsymbol{Gg} = \boldsymbol{0} & \boldsymbol{\lambda}(T) = \boldsymbol{0} \\ \dot{\boldsymbol{\lambda}} = \boldsymbol{G\lambda} & \boldsymbol{g}(0) = \boldsymbol{g_0} \end{cases} \tag{3.45}$$

In the system (3.45), the fourth equation is substituted in the first one, while from the second one the control vector $\boldsymbol{u}$ is evaluated:

$$\begin{cases} \boldsymbol{\lambda} = (\boldsymbol{G} + \boldsymbol{\phi_x}^T)^{-1} q_{\boldsymbol{x}} \\ \boldsymbol{u} = \boldsymbol{R}^{-T}\boldsymbol{B}^T \left[\Big((\boldsymbol{G} + \boldsymbol{\phi_x}^T)^{-1} \Big) q_{\boldsymbol{x}} \right] + \boldsymbol{u_T} \\ \dot{\boldsymbol{x}} = \boldsymbol{\phi} + \boldsymbol{Bu} - \dot{\boldsymbol{g}} - \frac{1}{2}\boldsymbol{Gg} = \boldsymbol{0} \\ \dot{\boldsymbol{\lambda}} = \boldsymbol{G\lambda} \end{cases} \tag{3.46}$$

By using the first one in the system above and by replacing $\boldsymbol{\lambda}$ in the fourth equation, it follows:

$$\left(\boldsymbol{G} + \boldsymbol{\phi_{x^T}}\right)^{-1} q_{\boldsymbol{xx}} \dot{\boldsymbol{x}} = \boldsymbol{G} \left(\boldsymbol{G} + \boldsymbol{\phi_{x^T}}\right)^{-1} q_{\boldsymbol{x}} \tag{3.47}$$

where it is imposed that: $\frac{d}{dt}\left(\boldsymbol{G} + \boldsymbol{\phi_x^T}\right)^{-1} q_{\boldsymbol{x}} \to \boldsymbol{0}$. And it gets:

$$\begin{aligned} \left(\boldsymbol{G} + \boldsymbol{\phi_{x^T}}\right)^{-1} q_{\boldsymbol{xx}} \dot{\boldsymbol{x}} &= \boldsymbol{G}(\boldsymbol{G} + \boldsymbol{\phi_x^T})^{-1} q_{\boldsymbol{x}} \\ \dot{\boldsymbol{x}} = \boldsymbol{\phi} + \boldsymbol{BR}^{-T}\boldsymbol{B}^T &\left[\left(\boldsymbol{G} + \boldsymbol{\phi_x^T}\right)^{-1} q_{\boldsymbol{x}} \right] + \boldsymbol{Bu_T} - \frac{1}{2}\boldsymbol{Gg} - \dot{\boldsymbol{g}} = \boldsymbol{0} \end{aligned} \tag{3.48}$$

As in the 1-D method, the second equation is substituted in the first one:

$$\begin{aligned} \left(\boldsymbol{G} + \boldsymbol{\phi_x^T}\right)^{-1} q_{\boldsymbol{xx}} \left[\boldsymbol{\phi} + \boldsymbol{BR}^{-T}\boldsymbol{B}^T \left[\left(\boldsymbol{G} + \boldsymbol{\phi_x^T}\right)^{-1} q_{\boldsymbol{x}} \right] + \boldsymbol{Bu_T} - \frac{1}{2}\boldsymbol{Gg} - \dot{\boldsymbol{g}} \right] = \\ = \boldsymbol{G}(\boldsymbol{G} + \boldsymbol{\phi_x^T})^{-1} q_{\boldsymbol{x}} \end{aligned} \tag{3.49}$$

The vector $\boldsymbol{g}$ is chosen so that $\boldsymbol{g} = [g; g; \ldots; g]$, and as in the mono-dimensional case, $g(t) = \text{cost} = g_0$. With these two new hypotesis, equation (3.49) becomes:

$$q_{\boldsymbol{xx}} \left[\boldsymbol{\phi} + \boldsymbol{BR}^{-T}\boldsymbol{B}^T \left[\left(g\boldsymbol{I} + \boldsymbol{\phi_x^T}\right)^{-1} q_{\boldsymbol{x}} \right] + \boldsymbol{Bu_T} - \frac{1}{2} g\boldsymbol{Ig} - \dot{\boldsymbol{g}} \right] = g\boldsymbol{I} q_{\boldsymbol{x}} \tag{3.50}$$

The generic nonlinear penalty function $\boldsymbol{q(x)}$ can be linearized as it follows:

$$\begin{aligned} q\left(\boldsymbol{x}\right) \approx& q\left(\boldsymbol{x_T}\right) + q_{\boldsymbol{x}}\left(\boldsymbol{x_T}\right)\left(\boldsymbol{x} - \boldsymbol{x_T}\right) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x_T})^T q_{\boldsymbol{xx}}\left(\boldsymbol{x_T}\right)\left(\boldsymbol{x} - \boldsymbol{x_T}\right) \\ q_{\boldsymbol{x}}\boldsymbol{(x)} \approx& q_{\boldsymbol{x}}\boldsymbol{(x_T)} + q_{\boldsymbol{xx}}^T\boldsymbol{(x - x_T)} \\ q_{\boldsymbol{xx}}(\boldsymbol{x}) \approx& q_{\boldsymbol{xx}}(\boldsymbol{x_T})^T = \boldsymbol{Q}^T \end{aligned} \tag{3.51}$$

The taylor expansion is substituted in (3.50):

$$\boldsymbol{Q}^T\left[\boldsymbol{\phi}+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[(g\boldsymbol{I}+\boldsymbol{\phi}_{\boldsymbol{x}^T})^{-1}\left(q_x(\boldsymbol{x_T})+\boldsymbol{Q}^T(\boldsymbol{x}-\boldsymbol{x_T})\right)\right]+\boldsymbol{B}\boldsymbol{u_T}-\frac{1}{2}g\boldsymbol{I}\boldsymbol{g}\right]= \\ g\boldsymbol{I}\left(q_x(\boldsymbol{x_T})+\boldsymbol{Q}^T(\boldsymbol{x}-\boldsymbol{x_T})\right) \tag{3.52}$$

The first order term $\boldsymbol{x}-\boldsymbol{x_T}$ is isolated on the right side of the equation, so:

$$\left[\boldsymbol{\phi}+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[\left(g\boldsymbol{I}+\boldsymbol{\phi}_{\boldsymbol{x}}^T\right)^{\boldsymbol{-1}}\left(q_x(\boldsymbol{x_T})+\boldsymbol{Q}^T(\boldsymbol{x}-\boldsymbol{x_T})\right)\right]+\boldsymbol{B}\boldsymbol{u_T}-\frac{1}{2}g\boldsymbol{I}\boldsymbol{g}\right]+ \\ -g\boldsymbol{I}\boldsymbol{Q}^{-T}q_x(\boldsymbol{x_T})=g\boldsymbol{I}\left((\boldsymbol{x}-\boldsymbol{x_T})\right) \tag{3.53}$$

Now, a Taylor expansion is made:

$$\begin{gathered}\left[\boldsymbol{\phi}+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[\left(g\boldsymbol{I}+\boldsymbol{\phi}_{\boldsymbol{x}}^T\right)^{\boldsymbol{-1}}\left(q_x(\boldsymbol{x_T})+\boldsymbol{Q}^T(\boldsymbol{x}-\boldsymbol{x_T})\right)\right]+\boldsymbol{B}\boldsymbol{u_T}+\right. \\ -\frac{1}{2}g\boldsymbol{I}\boldsymbol{g}-\boldsymbol{\frac{1}{2}}g\boldsymbol{I}\boldsymbol{g}-g\boldsymbol{I}\boldsymbol{Q}^{-T}q_x(\boldsymbol{x_T})\approx \\ \left[\boldsymbol{\phi}(\boldsymbol{x_T})+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[\left(g\boldsymbol{I}+\boldsymbol{\phi}_{\boldsymbol{x}}^T(\boldsymbol{x_T})\right)^{\boldsymbol{-1}}q_x(\boldsymbol{x_T})\right]+\right. \\ \left.+\boldsymbol{B}\boldsymbol{U}-\frac{1}{2}g\boldsymbol{I}\boldsymbol{g}-g\boldsymbol{I}\boldsymbol{Q}^{-T}q_x(\boldsymbol{x_T})\right]+ \\ +\left[\boldsymbol{\phi}_{\boldsymbol{x}}(\boldsymbol{x_T})+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[\nabla_x\left(g\boldsymbol{I}+\boldsymbol{\phi}_{\boldsymbol{x}}^T\right)^{\boldsymbol{-1}}_{\boldsymbol{x}=\boldsymbol{x_T}}q_x(\boldsymbol{x_T})\right]+\right. \\ \left.+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[\left(g\boldsymbol{I}+\boldsymbol{\phi}_{\boldsymbol{x}}^T(\boldsymbol{x_T})\right)^{\boldsymbol{-1}}\boldsymbol{Q}^T\right]\right](\boldsymbol{x}-\boldsymbol{x_T})\end{gathered} \tag{3.54}$$

and now two equations can be derived: one is dependent from the state, one independent from the current state:

$$\begin{aligned}\boldsymbol{G}=&\left[\boldsymbol{\phi}_{\boldsymbol{x}}(\boldsymbol{x_T})+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[\left(g\boldsymbol{I}+\boldsymbol{\phi}_{\boldsymbol{x}}^T(\boldsymbol{x_T})\right)^{-1}\boldsymbol{Q}^T\right]\right] \\ \boldsymbol{B}\boldsymbol{u_T}=&\left[\boldsymbol{\phi}(\boldsymbol{x_T})+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[\left(g\boldsymbol{I}+\boldsymbol{\phi}_{\boldsymbol{x}}^T(\boldsymbol{x_T})\right)^{-1}(q_x(\boldsymbol{x_T}))\right]-\frac{1}{2}g\boldsymbol{I}\boldsymbol{g}-g\boldsymbol{I}\boldsymbol{Q}^{-T}q_x(\boldsymbol{x_T})\right]\end{aligned} \tag{3.55}$$

It is reasonable to assume that $q_x(\boldsymbol{x})\to 0$ when $\boldsymbol{x}\to\boldsymbol{x_T}$, so:

$$\begin{gathered}\boldsymbol{G}=\left[\boldsymbol{\phi}_{\boldsymbol{x}}(\boldsymbol{x_T})+\boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^T\left[\left(\boldsymbol{G}+\boldsymbol{\phi}_{\boldsymbol{x}}^T(\boldsymbol{x_T})\right)^{-1}\boldsymbol{Q}^T\right]\right] \\ \boldsymbol{B}\boldsymbol{u_T}=\frac{1}{2}\boldsymbol{G}\boldsymbol{g}-\boldsymbol{\phi}(\boldsymbol{x_T})\end{gathered} \tag{3.56}$$

And the final form is:

$$
\begin{aligned}
\dot{\boldsymbol{x}} &= \boldsymbol{\phi} + \boldsymbol{B}\boldsymbol{u} - \frac{1}{2}\boldsymbol{G}\boldsymbol{g} \\
\boldsymbol{u} &= \boldsymbol{R}^{-T}\boldsymbol{B}^{T}\left[(\boldsymbol{G} + \boldsymbol{\phi}_{\boldsymbol{x}}^{\boldsymbol{T}})^{-1} q_{\boldsymbol{x}}\right] \\
\boldsymbol{B}\boldsymbol{u}_{\boldsymbol{T}} &= \frac{1}{2}\boldsymbol{G}\boldsymbol{g} - \boldsymbol{\phi}(\boldsymbol{x}_{\boldsymbol{T}}) \\
\boldsymbol{G} &= \boldsymbol{\phi}_{\boldsymbol{x}}(\boldsymbol{x}_{\boldsymbol{T}}) + \boldsymbol{B}\boldsymbol{R}^{-T}\boldsymbol{B}^{T}\left[(\boldsymbol{G} + \boldsymbol{\phi}_{\boldsymbol{x}}^{\boldsymbol{T}})^{-1}\boldsymbol{Q}^{T}\right]
\end{aligned}
\tag{3.57}
$$

Also in the multidimensional case now discussed, the method succeeds in giving an initial form for the $\boldsymbol{G}$ matrix. This form of the matrix is different from the one discussed in the intuitive presentation, since the one in the previous section was a diagonal matrix of the form $\boldsymbol{G} = -\frac{\boldsymbol{I}}{\Delta_T}$, while the one derived in equation (3.57) is not diagonal in general. Nevertheless, this does not mean that the diagonal form can not be used as a starting condition for the $\boldsymbol{G}$ matrix, which is simple to test and it has the great advantage of require just one tuning parameter. One of the possible way to simplify the structure derived in this paragraph is to apply the least minimum square technique on the $\boldsymbol{G}$ matrix and having it diagonal.

Chapter 4

Application of FLOP method in prototype problems

In the previous chapter the FLOP technique has been explained starting from the initial idea, then with a rigorous approach both in the monodimensional and in the multidimensional cases. It has been shown that FLOP control can be derived starting from a variational problem, using an adjoint variable to the classical problem. In this chapter, two prototype problems are shown: the first one is referred to a linear dynamical system with quadratic cost function both in the state variable and in the control one. The test case is chosen monodimensional. The second prototype problem is a nonlinear multidimensional case, in which the FLOP method is compared to different control algorithms, the State Dependent Riccati Equation (SDRE), the Model Predictive Control (MPC), the Nonlinear Model Predictive Control (NMPC) and the Linear Quadratic Regulator.

4.1 Monodimensional case

In this case the dynamical system is chosen as $\dot{x} = \tanh(x) + u$, with the cost function $E(x,u) = \frac{1}{2}qx^2 + \frac{1}{2}ru^2$, so the statement of the problem is:

$$\min \bar{J} = \int_0^T \left[\frac{1}{2}qx^2 + \frac{1}{2}ru^2 + \lambda(\dot{x} - (\tanh(x) + u))\right] \tag{4.1}$$

In this first prototype case the FLOP method is tested against one of the most classical techniques for solving control problems, the Linear Quadratic Regulator [54]. Although the comparison seems unfair, since the LQR technique works with linear dynamical system with quadratic cost functions, it is not unusual to see LQR applied also in nonlinear cases, using a linearization of the dynamical system. Here, the LQR is therefore linearized on the target point $x_T = 2$, with starting position for the state variable placed in $x_0 = -5$. The LQR needs to rewrite the system as:

$$\begin{aligned} \dot{x} =& a(x - x_T) + u_{LQR} - u_T \\ &\text{with} \\ a =& \tanh(x_T) - x_T(\tanh(x_T)^2 - 1) \end{aligned} \tag{4.2}$$

while, following the standard procedure for the FLOP technique, its control is:

$$u_{FLOP} = \frac{q(x - x_T)}{r\Big(\tanh(x) - x(\tanh(x)^2 - 1) + G\Big)} - x_T \tanh(x_T) \tag{4.3}$$

In Figure 4.1,4.2 and 4.3 the state, control and the cost function both for the FLOP and the LQR methods are shown. As in following figures, the FLOP method suceeds in reaching the target before the LQR technique, and the control can take account of the nonlinearity around the value $\bar{x} = 0$.

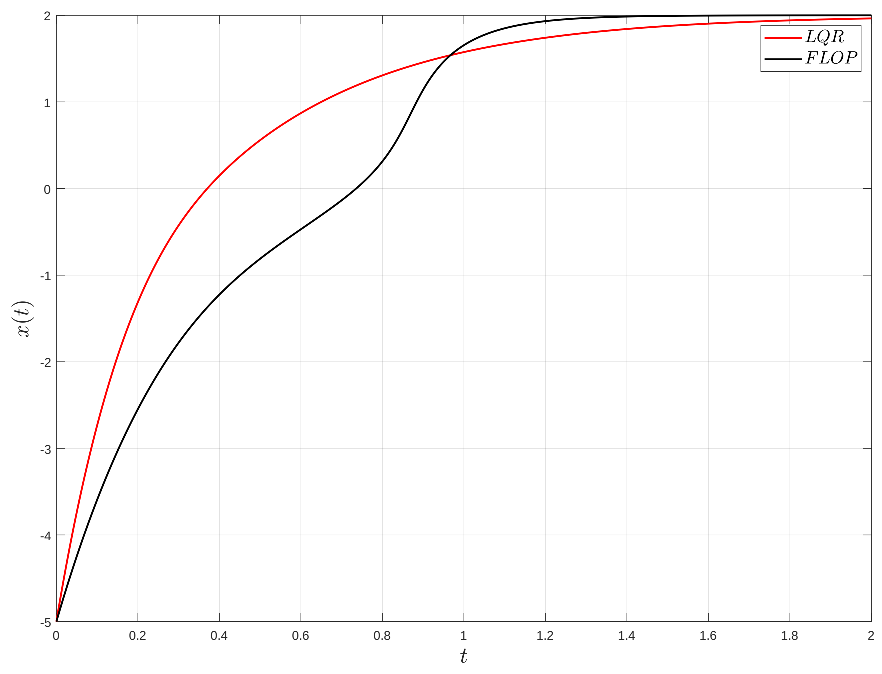

Figure 4.1. State variable for both FLOP and LQR control

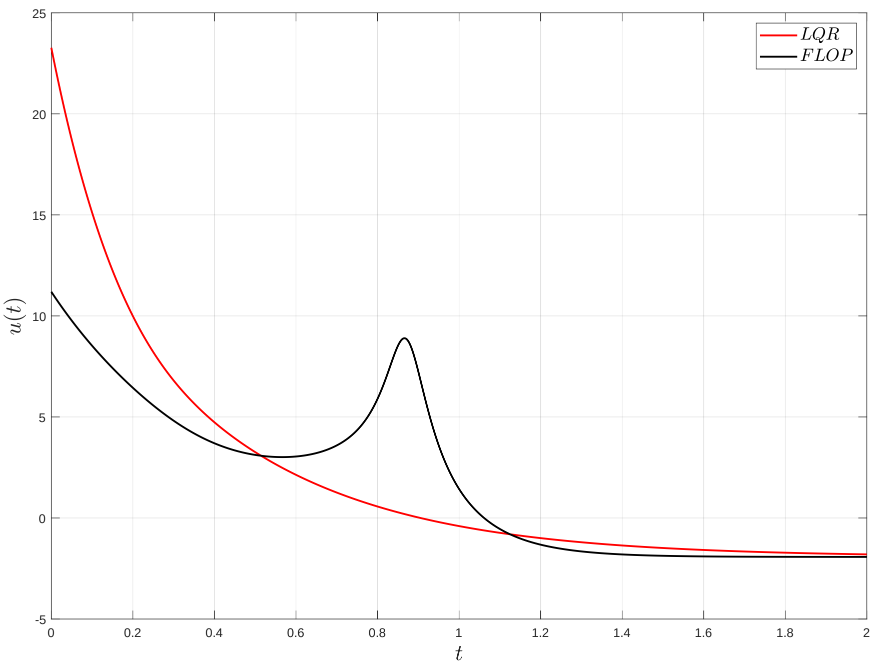

Figure 4.2. Control variable for both FLOP and LQR control

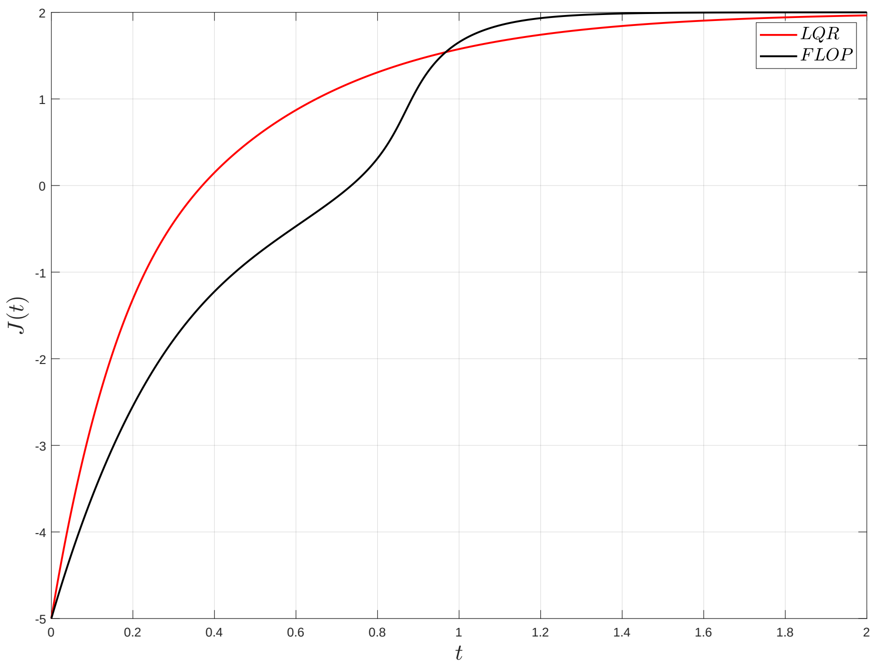

Figure 4.3. Cost function for both FLOP and LQR control

The LQR, since is linearized in the target position x_T can not take account of the nonlinearity but it also requires an higher cost function. This first prototype case showed the potential of the FLOP method in working with nonlinear dynamical system. In this case the comparison with the LQR is almost unfair, since it is not able to consider the nonlinear part of the dynamical system. Nevertheless, it is not unusual the try to control nonlinear systems with the linearized LQR technique, and in any case the LQR here has the objective to be a benchmark for the Feedback Local Optimality Principle. In the next example the FLOP method is tested in a multidimensional case with some well-known techniques, which are more related to nonlinear dynamical systems, as the SDRE and the NMPC.

4.2 Multidimensional case

In this section the FLOP method is tested on a nonlinear multidimensional dynamical system. To test its effectiveness it is compared to several well-known techniques (SDRE [48], MPC [63] , NMPC [64], LQR). The test case is the system:

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_1 x_2^2 + (1+\sqrt{|x_1|})(2u+u^2)\end{aligned} \tag{4.4}$$

where the state variable is $\boldsymbol{x} = [x_1; x_2]$ In this test case, the purpose of the control is to stabilize the system in the reference point set at $P = [0; 0]$. As it can be easily seen, in P the control value is $u = 0$. Furthermore, the functional J has to minimized, where:

$$J = \frac{1}{2}\int_0^\infty (2x_1^2 + x_2^2 + u^2)dt \tag{4.5}$$

The algorithms are here compared in three different cases, each one of them characterized by different initial conditions for the sistem, i.e. $x_{01} = [1; 0]$, $x_{02} = [.5; 0]$, $x_{03} = [.5; .5]$. To apply the SDRE technique is needed a quasi-linearization, so the system is rewritten in the form: $\dot{x} = A(x)x + \bar{B}(x,u)u$:

$$\begin{bmatrix}\dot{x}_1 \\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1 \\ x_2^2 & 0\end{bmatrix}\begin{bmatrix}x_1 \\ x_2\end{bmatrix} + \begin{bmatrix}0 \\ (1+\sqrt{|x_1|})(2+u)\end{bmatrix} u \tag{4.6}$$

In this case, the system may be uncontrollable for the control value $u = -2$, since it causes the matrix $\bar{B}(x,u)$ to be $[0; 0]$.
The dynamical system is written in the form $\dot{x} = \phi(x) + B(x,u)$, therefore the FLOP method can not be applied, since it requires the dynamical system to be linear in the control variable u, and not dependent from the state variable. Therefore, a linearization of the term $B(x,u)$ must be performed. It is chosen to linearize it around the reference point P. Since the gradient of $B(x,u)$ is requested, this choice of the linearization in P can not be performed, since the gradient in P is undetermined. Hence, a neighbour point $\hat{P} = [.01; .01]$ is chosen as a linearization point. Also the Linear Quadratic Regulator technique and the Model Predictive Control require a linearization of the dynamical system, which has also been linearized in $\hat{P}$. In following figures the comparison is shown in terms of the state variables, the control variable and the cost function, which are also summarized in the following table.

Looking at the results, different aspects must be highlighted: SDRE, MPC and LQR techniques have similar response in the state variable x_1, while some differences appear regarding the NMPC and the FLOP methods: The NMPC reaches faster the target with respect to the FLOP method, which is more gentle and reaches the target smoothly. Nevertheless, the FLOP approach still is faster than LQR, SDRE and MPC methods to reach the target for the state x_1. In the state x_2 the FLOP approach presents important differences between its behaviour and the other technique's one. Even in this case the NMPC appears to be the one with faster response to reach the target, with the FLOP method having a smoother evolution. Regarding the control request, SDRE and LQR are the two techniques which demand the higher magnitude of control. The MPC and the NMPC present similar behaviour, while the FLOP approach appears to be the less demanding in terms of the control effort with the chosen parameter values.

Initial Conditions	[1;0]	[.5;0]	[.5;.5]
Method	J	J	J
FLOP	**1.086**	**0.2663**	**0.628**
SDRE	1.157	0.2675	0.8632
MPC	1.1	0.2872	0.7961
NMPC	1.079	0.2677	0.6065
LQR	1.763	0.3679	1.2726

Chapter 5

Swarm Robotics application

In this chapter, an introduction to the swarm robotics and its state of the art is performed. Then, one application in the marine environment is shown. In particular, the Feedback Local Optimality Principle is tested on the control of a swarm of marine drones for the rescue of a breakdown boat, i.e. a former member of the swarm. Different rescuing strategies will be tested, proving the ability for the FLOP control to manage different agents to move and complete tasks in a complex environment.
Swarm robotics is the study of how is it possible to make robots collaborate and solve a task as a collective. In general, the required task would be impossible to solve by a single individual of these robots. Therefore, the aim of swarm robotics is the collaboration between agents and their teamwork.
First, it is necessary to define a swarm, but despite the huge amount of literature present nowadays, a proper definition of the swarm is still missing. There are many definitions about the behaviour of the swarm, and in particular in English there are many words for a swarm behaviour, depending on what are the elements of the swarm itself: for example, it is named flock in the case of birds, school in the case of fish, herd in the case of quadrupeds [65, 66].
Without giving a proper definition of the swarm, the most important aspect is to define how many elements compose a swarm. Beni [67, 68], in an accepted definition from the scientific community, says that the swarm size is defined from what is not: "not as large as to be dealt with statistical averages" and "not as small as to be dealt with as a few-body problem". It is generic, but it gave a reasonable idea of how many elements are necessary to define a proper swarm. The 'definition' is clear with large numbers, but of course its lacking of information in the lower bound: of course one agent is not a swarm, and for sure two agents can not form a swarm. Though, depending on the problem, with $N = 3$ agents it is probable that agent start form swarm behaviors, as it will be shown in the example of rescuing a boat.
Once the concept of swarm is cleared, the goal is to proper define the swarm robotics. Dorigo and Sahin [69, 70] defined it as: *"Swarm robotics is the study of how a large number of relatively simple physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment"*. The concept of relatively simple robots should be investigated, but for this work we can consider the simple robot as an agent incapable of completing the assigned task alone. In swarm robotics, local

interaction between agents and between agents and the environment are fundamental. This relies in the necessity of create a collaboration between agents and a global knowledge on the environment for the swarm. As an additional definition, Beni defines an intelligent swarm as a "group of non-intelligent robots forming, as a group, an intelligent robot".

5.1 Generalized characteristics for swarm control

Swarm robotics presents interesting advantage in modern engineering, and the first one is due the possibility to build simple (and low-cost) robots which can complete different tasks, which without the concept of swarm robotics may be completed by a very complex (and therefore high-cost) robot. In general, three main advantages [71, 72] are known when the swarm robotics is used, and therefore these three advantages are requested in a swarm robotics problem:

- Robustness
- Flexibility
- Scalability

The first point, the robustness, is related to the fail safe. Giving a complex mission to a high number of robots means that even if some of the robots can not complete the mission, this may be accomplished by other robots. This would not possible if the mission was assigned to a single robot, and the probability for the mission to fail would be higher.
The concept of flexibility is aimed to the fact that robots in a swarm are interchangeable. There are no specific task for each robot, but the swarm as a group has to complete an assigned mission. Moreover, when one robot is not able to complete the task, other robots can help the first one. This is related for example to the moving of a pack using a robot swarm. If the single robot has not enough actuation to move the pack, other robots can push together with the first one and the sum of their actuation can provide the job to be done.
Since each agent in general communicate only with its neighbours, task can be completed even if the number of agents increases, since robots will communicate and they will maintain a chosen density around the important points of the environment.
When speaking of swarm robotics, there are always two ways to look at the problem: the micro (local) point of view and the macro (global) one. When looking at the microscopic level, the focus is in the behaviour of the single agent. For example, it has to be observed if the single agent is avoiding crashes with all other agents, or crashes with obstacles in the environment. Another example is the observation of the forces exchange between one agent and another, or between one single agent and the pack to move. On the macroscopic level, the focus is on the collective behaviour of the swarm and in the success or the failure of the global task of the swarm. For example, in the case of moving an object from one place to another through a robot swarm, from a macroscopic level is not important how many robots are actually pushing/pulling the object to the target, but it is important to see if the object is moving to the target. In other words, the task of the swarm is designed on the

macroscopic level, while the implementation of the control is made on the single agent (microscopic) level.
In swarm robotics there are some well-defined tasks, and in general any application in this field can be represented as a combination of more of these tasks. For this reason, here this basics tasks are introduced:

- **Aggregation**
  Aggregation [73] is maybe the most fundamental behavior for swarms, since for the intrinsic definition of swarm is important for the agents to stay together. It is observed even in the natural world, and several examples come in mind: honeybees, ladybugs, butterflies, birds, fishes. The goal in the aggregation task is to minimize the relative distance between robots. The target position around which the aggregation must be performed can be defined or unspecified. The aggregation task can be performed in different ways: it can be asked to each agent to go as near as possible to a chosen location, or it can be asked to each agent to randomly move until they are close to another agent, and then to stop. Clearly these two possible strategies can lead to different aggregation form for the swarm, therefore one main issue is the well-definition of the problem.

- **Dispersion**
  Dispersion [74, 75] is the task opposite to the aggregation. One way to see this task it to ask to each robot to place itself as far as possible from any other robot. Since this would lead to an infinite motion of the robots which would tend to occupy the entire space, in general it is requested to the robot to place itself as far as possible from all other robots, but also to maintain the communication active. The communication between robots is in general limited by the sensors they have, so these two requests will lead the swarm to find an equilibrium.

- **Pattern formation**
  The pattern formation can also be seen as an extension of the aggregation and the dispersion tasks, but it give less freedom to the agents and it has more strictly requirements[76–78]. As examples, the trail networks formed by ants is a pattern formation, and so it is the traffic flow in pedestrians, where people walk in one direction if they are on one side of the road and in the opposite direction if they are on the other side of the road.

- **Collective construction**
  This task is in general considered much more complex than the previous ones. It involves great organization and communication between agents, it can be performed at very different levels: one possibility is having each robot with a precise 'block' to move and to place in a precise position. Another possibility is having each robot with the task to pick the closest 'block' and to place it in the nearest empty position of the construction. As it can be imagined, the second strategy is much more complex than the first one, even if it is probably much more efficient.

- **Collective transport**
  This scenario is very promising for swarm robotics, because it is possible that

some kind of swarm effects can be seen [79, 80]. The main reason is in a problem of an heavy object which must be moved by the swarm, but one agent is not able to push the object if alone. In this case, there should be a certain threshold in terms of swarm size for moving the object. In next section an application for this tasks is shown: a swarm of marine drones must rescue a breakdown boat an take it to a safe area.

- **Flocking and collective motion**

Flocking is one of the names in which the coordinated motion of robots is identified. In the natural world is referred to bird 'groups', and it regards the possibility for a swarm to move in the same direction with approximately the same speed. Reynolds [81] provide three rules which are able to emulate a flock of birds:

1. Adapt your direction to that of your neighbors
2. Stay close to your neighbours
3. If you are too close, avoid the collision

In next section, an example of a unicycle swarm of robots migrating in a complex environment is shown, based on a model similar to the Vicsek [82] one.
The Feedback Local Optimality Principle has been proved to be very useful in swarm robotics applications, due its flexibility, robustness and low computational cost.
The possibility to use any nonlinear dynamical system and the chance to have non-quadratic cost functions permits to impose different behaviors to the swarm. Moreover, even if each agent is controlled individually, it is possible to create through the FLOP method a collective behaviour for the entire swarm. In this sense, in the next section two different applications in the field of swarm robotics are proposed: the first one regards what is called the migration problem, and it belongs to the flocking and collective motion task. A swarm of unicycle robots has to move from one area to another of an unknown environments; the second one regards the rescue of a breakdown boat through a swarm of marine drones. This application belongs to the collective transport scenario, and it shows some interesting capabilities obtained with a swarm, even with a small number of agents.

5.2 Unmanned Vehicles

As fundamental part of swarm robotics, it is surely important to focus the attention on Autonomous Vehicles (AVs). One of the possible classifications for autonomous vehicles [83, 84] is the one related to their environment, such as:

- Underwater robots, usually called Autonomous Underwater Vehicles (AUVs)
- Boat robots, Unmanned Surface Vehicles (USVs)
- Aerial drones, Unmanned Aerial Vehicles (UAVs)
- Ground robots, Unmanned Ground Vehicles (UGVs)

In each of the cited categories, different applications and different kind of robots can be found in literature, and it is almost impossible to cover all the different projects / research.
Nevertheless, one of the many interesting facts on AVs regards the interconnections between different categories: some problems are intrinsically tied to the concept of AVs. In these common problems to all categories, obstacle avoidance is surely one of the main aspect which needs to be studied. For instance, the problem of obstacle avoidance is not 'simply' the factual avoidance of an obstacle on the path, but it involves many capabilities for the robot, from high to low engineering level (perception of the obstacle, sensing, decision making, actuating, safety, etc.). These capabilities does have a small dependence on which category the robot belongs. In this section, the chance to implement obstacles avoidance technique in AVs through the Feedback Local Optimality Principle is taken. Three different robots (USV, UAV, UGV), are here presented. Each robot is modelled so that the Feedback Local Optimality Principle can control it, and for each robot an application for the obstacle avoidance is presented. The aim of this section is to show the flexibility and the robustness of the FLOP control with different dynamical systems put in different scenarios, but whit a common task, crucial for any autonomous vehicle.
Safety First
In all Unmanned Vehicles categories, safety is one of the crucial aspects in the current research. In fact, according to [85], 93% of road accidents are caused by human error and even with a small introduction of AVs this percentage could be lowered down sensibly. Actually, some studies [86–89] sustains the full presence of autonomous vehicles will reduces insurance costs by around 90%. This could not be completely true, since with AVs different new risk can be introduced:

- Electronic systems can fail, and so can sensors. In difficult conditions a small failure can create huge damages.

- Any technology can be hacked for crime purposes.

- Difficulties in detecting and communicating with bicyclists, pedestrian, motorcycles. One of the main aspects in safety is for the autonomous vehicles to have a reliable, safe and tested obstacle avoidance strategy.

5.2.1 Obstacle Avoidance Techniques

The problem of the mobile robot navigation refers to the capability of the robot to safely move in the direction of the target, using its knowledge and all information it gets from the surrounding environment. Usually, these are on going information, because the knowledge of the environment is partial or, in some cases, completely absent. Moreover, the environment is non-static, which means during its path the robot can collide with other robots. Hence, for a safe navigation, in addition to the path planning it is necessary a local procedure for the obstacle detection and the obstacle avoidance. In this section, the path planning strategy is assumed given, while some obstacle avoidance techniques are illustrated.

Bug Algorithms

Bug Algorithms (BAs) are simple methods [70, 90] to avoid unexpected obstacles from a starting point S to a target point T. The task of these algorithms is to generate a free-collision path from S to T. There are many different bug algorithms, but they all start with two different capabilities for the robot:
- follow a wall (right or left)
- move in a straight line toward goal
The advantages of bug algorithm are in general two:

1. They assume only *local* knowledge of the environment and a global goal.

2. In any case they always succeeds, if it is possible, in reaching the goal.

Of course, BAs have some drawbacks, first of all the inefficiency in the path. Two main algorithms are here cited:

- **Bug Algorithm 1**: whenever the robot sense an obstacle, the robot, starting from the detection point H, goes around the obstacle until it returns in H, with the 'follow a wall' strategy. Since it can measure the distance between itself and the target T, after the first whole lap, it returns in the point L where the minimum distance from the target was detected. From there, it uses the second strategy and it moves in a straight line toward the goal.

- **Bug Algorithm 2**: as in the first algorithm, whenever the obstacle is detected the robot starts to move around the obstacle. In this case, the robot ends the follow a wall strategy in the moment where is aligned with the segment that unites the starting point S with the target point T.

These algorithms are easy both to understand and to implement, but they do not account the kinematics of the robot, which is a strong limitation expecially with non-holonomic robots.

Potential Field Methods

The path planning methods whose use these strategies, are based on a simple and powerful principle which has the intrinsic ability to avoid obstacles. The robot is considered as a particle which moves in a potential field generated by the target and the obstacles in the environment. The target produces an attractive potential, while obstacles produce repulsive potential. The obstacles can be both known *a priori* or can be detected by sensors on board. The gradient of the potential field generates an attractive force (toward the goal) and repulsive forces (caused by the obstacles). For obstacle detected by sensors, in general it is used the *Virtual Force Field (VFF)* [91, 92].

Virtual Force Field

The environment is represented with a bi-dimensional grid C [93]. Each cell in the grid has a value V, which depends on the certainty/uncertainty of the possibility to find an obstacle. The higher the value, the higher the possibility to have an obstacle in that cell. To evaluate the repulsive force, a sub-grid C^* is generated with the robot placed in the center of it. Each cell of C^* express a repulsive force proportional to the value V and from the square distance between the robot and the cell. The attractive force is always accounted and it depends on the distance between the robot and the target. A simple analogy can be made by thinking to the robot as a positive charged particle, the target as a negative charged particle, all obstacles are positive charged particles. The combination of attractive and repulsive forces guide the robot in a safe path towards the target. Identifying the position f the robot with q, the potential field is represented by $U(q)$ and it is the sum of all repulsive and attractive forces. The particle tries to reach the minimum of the potential U, and the forces which guide it are represented by the gradient of U, represented by ∇U. One used example is to use the velocity vector proportional to the force of the potential field, so that the robot moves with a velocity which decreases when the robot is closer to the target q_T. For example,

$$\begin{aligned} F_{att}(q) &= -\nabla U_{att}(q) = -k_{att}(q - q_T) \\ F_{rep_i}(q) &= \begin{cases} k_{obs_i}\left(\frac{1}{d_{obs_i}(q)} - \frac{1}{d_0}\right)^2 & \text{if } d_{obs_i}(q) < d_0 \\ 0 & \text{if } d_{obs_i}(q) > d_0 \end{cases} \end{aligned} \tag{5.1}$$

The main problem with these kind of algorithms is represented by the local minima. In fact, it is possible that in some points the attractive force and the repulsive forces are equals, leading to a local minimum such that $\nabla U(q^*) = 0$. This is unfortunately usual with convex obstacles, but it can happen also with non-convex obstacles. To avoid this problem, the repulsive force can be considered dependent not only from the distance to the obstacle but also from the heading of the robot with respect to the obstacle. This is reasonable, since it is less urgent to avoid an obstacle if it is parallel to the current velocity of the robot with respect to an obstacle in front of the robot.

Vector Field Histogram

The Vector Field Histogram (*VFH*) method is similar to the *VFF*, so there is a bi-dimensional grid C which models the surrounding environment. When an obstacle has been detected in the $i-th$ cell, value V_i increases. Differently from the *VFF*, the *VFH* method provides a reduction of data to evaluate the control action desired. In the first phase, a sub-grid C^* is individuated. The histogram is then divided into angular sections. Each *j-th* section has a value V_{S_j} which represents the obstacle's density in the section, evaluated as the sum of the values V_i of each cell in the sub-grid C^*, weighted on the distance between the cell and the robot.
The second phase search for the close sector to the desired direction with a mean value $V_{S_j} < V_{th}$, with V_{th} chosen threshold value. The threshold is a tuning parameter and the decision of its value influences highly the success or failure of the method. In fact, if the threshold is too low, this means the robot will not be able to pass in narrow passages, while if the threshold is too high the robot will collide into some obstacles. The *VFG* has been developed into two extended algorithms, the *VFH*$^+$ and the *VFH**.

Vector Field Histogram$^+$

In this algorithm two main adjustment are performed:
- the first one is that the robot dynamics is considered - the second one is that each obstacle is enlarged of an angle γ known *a priori*. With the first assumption, it is assumed that the robot can perform straight trajectories and circular arc trajectories, so that most robots can assure the success in the trajectory to follow. The minimum radius of steering are function of the velocity of the robot and can be adjusted depending on the robot dynamics. The second assumption is made to provide an ulterior safe parameter to decrease as much as possible the collision's probability. The resulting histogram is known as the *masked polar histogram* which choose the possible trajectory at the current velocity.

Vector Field Histogram*

Once again, one of the main problem with the *VFH*$^+$ method is that it can be stuck in some local minima. The *VFH** provides an adjustmen to the local minima problem by using the resulting histogram provided by the *VFH*$^+$ as prediction steps for a small interval *dt*. Than, from each new possible position another *VFH*$^+$ is performed and so on, in a recursive manner. So, different possible paths are analyzed and at the end it is chosen the path with lower costs.

Obstacle Restriction Method

The main difference between the Obstacle Restricion Method (*ORM*) and the ones previously mentioned is that *ORM* uses all the information on the obstacles in each step, therefore there is no reduction on data. The method requires obstacles are in the form of points (obstacle points), and it is required a sensor able to sense continuously the distance between the obstacle and the robot. This algorithm can be divided into two parts:

1. Evaluation of possible directions

2. Selection of the better direction

During the first part, clearly one of the possible directions is the one which unites the robot with the target location. Since it is not always possible to reach directly the target, a collection of sub-targets are created. A sub-target is created: - in the space between two obstacle points, whose distance must be greater than the diameter of the robot.
- in the direction of the frontier of an obstacle, at a distance greater than the diameter of the robot.
Once the list of target/sub-targets is completed, the decision of which direction the robot must follow is made. If the robot can reach directly the target, that is the chosen direction. Otherwise, it is chosen the sub-target with the minimum angular distance from the the angle which identifies the direction towards the target.

Dynamic Window Approach

This algorithm determines the couple of linear and angular velocities $V_T = (v, \omega)$ of the robot. To do so, starting from the current linear and angular velocities of the robot, it considers three different ranges of possible couples:

- V_s : the couples of admissible velocities that the robot can reach
- V_a : the couples of admissible velocities the robot can reach before hitting an obstacle
- V_d : the couples of admissible velocities considering the linear and angular accelerations $\dot{v}, \dot{\omega}$.

Then, the target couple of velocities is determined by

$$V_T = (v, \omega) = \max \Big[\sigma(\alpha H((v, \omega)) + \beta D(v, \omega) + \gamma V(v, \omega))\Big] \tag{5.2}$$

where H, D, V represents the angle between the current heading the direction towards the target, the distance between the nearest obstacle and the robot, and the velocity of the robot. α, β, γ are the weights of the three terms, which are dependant on V_s, V_a, V_d.

Velocity Obstacle

The Velocity Obstacle (VO) algorithm identifies the collection of all velocities of one robot which lead to the collision with a second robot, supposing the second robot maintains its velocity constant. Considering two supposedly circular robots, A and B, with respective velocities v_A, v_B. To construct the VO algorithm, A is reduced to a point $\hat{A}$ and B is enlarged of the A radius, becoming $\hat{B}$. A collision cone is defined as:

$$CC_{A,B} = v_{A,B} = v_A - v_B \mid \lambda_{A,B} \cup \hat{B} \neq \emptyset \tag{5.3}$$

Each velocity which follow inside the cone will lead to a collision. If more than one obstacle has to be considered, the resulting area will be the union between all the possible areas, so that:

$$VO = \bigcup_{i=0}^{m} V_{B_i} \tag{5.4}$$

where V_{B_i} are the cones generate by the m - obstacles. In the case of more than one obstacle, it is reasonable to give a high priority to closer obstacles. This is for many reasons: one is the 'safety first' principle. Secondly, the VO is based on a linear approximation of the obstacle trajectory, so it will be more precise in closer possible collisions than farther one. One of the problem is to define the concept of "imminent collision", and this is solved by identifying an imminent collision as the one which occurs in a time $t < T_{th}$, with T_{th} which has to be decided depending on the velocities and accelerations of the robot.

5.2.2 FLOP applicability in the Obstacle Avoidance technique

Due the importance of the Obstacle Avoidance for any unmanned vehicle's application, in this section is explained how the Feedback Local Optimality Principle can provide a reliable obstacle avoidance technique, starting from the ones described above.
The intrinsic construction of the FLOP as a variational based method, suggest the possibility to use a model for the obstacle avoidance technique close to the Potential Field Methods. In fact, in the original formulation of the FLOP control it appears the potential function $g(\boldsymbol{x})$, which is a state dependant potential function. The FLOP formulation is here rewritten for simplicity:

$$\min \bar{J} = \int_0^T g(\boldsymbol{x}) + \boldsymbol{u}^T \boldsymbol{R} \boldsymbol{u} + \boldsymbol{\lambda}^T (\dot{\boldsymbol{x}} - (\boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{B}\boldsymbol{u}))dt \tag{5.5}$$

The chance to use highly nonlinear potential functions in the cost function can provide to the FLOP method the possibility to add a series of 'task' to the unmanned vehicle, including the obstacle avoidance one.
It is here important to denote that the chance to add different nonlinear functions can make high difference in the success or failure of the obstacle avoidance technique, and the choice of a function instead of another can change drastically the behavior of the unmanned vehicle. For instance, if a quadratic repulsive function is added, centered on the obstacle, the vehicle will sense the obstacle in all the environment. This, of course, is not what is searched for the obstacle avoidance technique. The goal is to have a function which can mimic the actual behavior of an obstacle avoidance. To implement an obstacle avoidance technique which can be used in actual engineering applications, the choice is to use Gaussian-like functions, such as:

$$g(x) = \frac{K}{\sigma\sqrt{2\pi}} \exp -\frac{(x - x_{obs})^2}{2\sigma^2} \tag{5.6}$$

where K, σ, x, x_{obs} are the amplitude and the variance of the gaussian function, the position of the vehicle and the position of the obstacle, respectively. or in the two-dimensional case:

$$g(\boldsymbol{x}) = K \exp \Big[- \Big(\frac{(x - x_{obs})^2}{2\sigma_x^2} + \frac{(y - y_{obs})^2}{2\sigma_y^2} \Big) \Big] \tag{5.7}$$

The reasons behind this choice are mainly:

- Gaussian-like functions have only two parameters to be tuned, the amplitude K and the variance σ. This helps the tuning phase, which is always a fundamental part of any engineering application.
- Gaussian-like functions can be very wide or very peaked, and this is useful since it is important to have different possibilities to use in different situations. In fact, with just the two parameters the shape of the obstacle can be mimicked. In multi-dimensional cases, the obstacle can be represented by varying the variance-covariance function.
- This kind of function can be used also in the case of the Internal Avoidance penalty function, by substituting the obstacle coordinates with the coordinates of a second autonomous vehicle.

The Internal Avoidance penalty function is the function which permits to different agents to not collide. It has to be intended as an obstacle avoidance penalty function, with a 'moving obstacle', which is a second agent. Here, some examples are made. In the Figure 5.1 an environment with one obstacle in the position $\boldsymbol{x_{obs}} = [-4; -4]$ is shown.

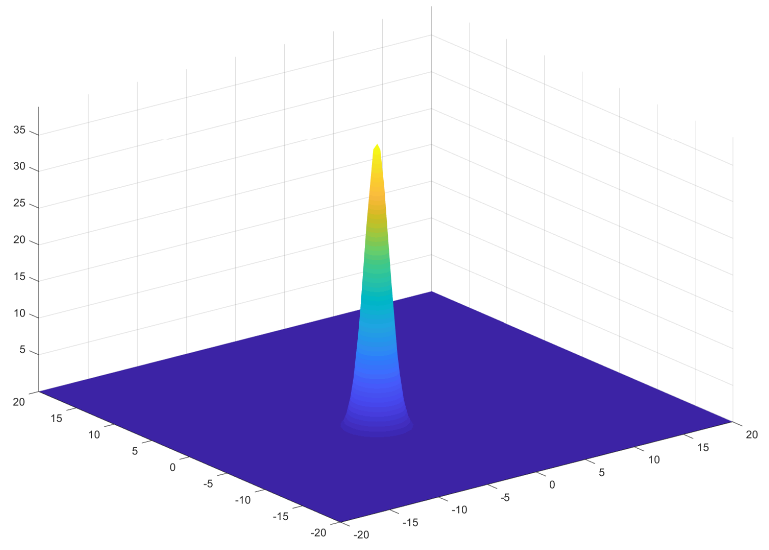

Figure 5.1. Map of the Environment with one obstacle

The $g(\boldsymbol{x})$ function gives also the possibility to add different obstacles of different sizes and shapes, as shown in Figure 5.2

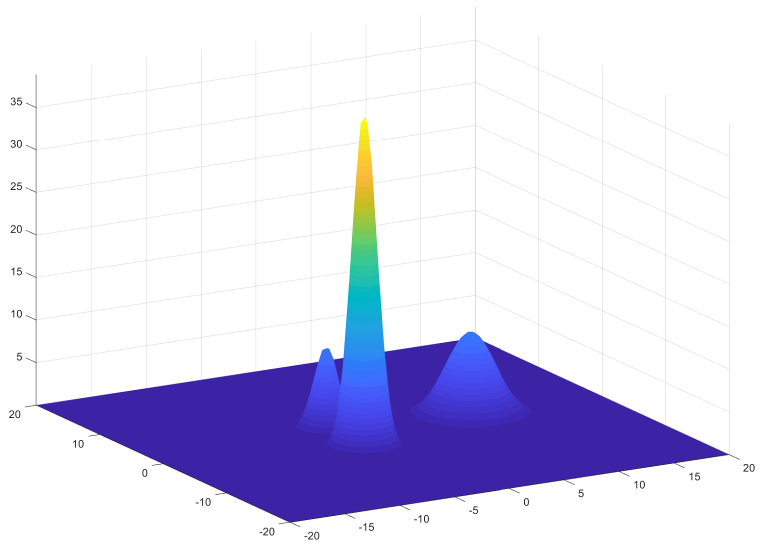

Figure 5.2. Map of the Environment with different obstacles

Lastly, the $g(\boldsymbol{x})$ is useful to give also all other tasks to the unmanned vehicles, such as the 'go to target' one. This last task can be represented as an attractive potential centered in the position of the target $\boldsymbol{x_T}$, as shown in Figure 5.3 below:

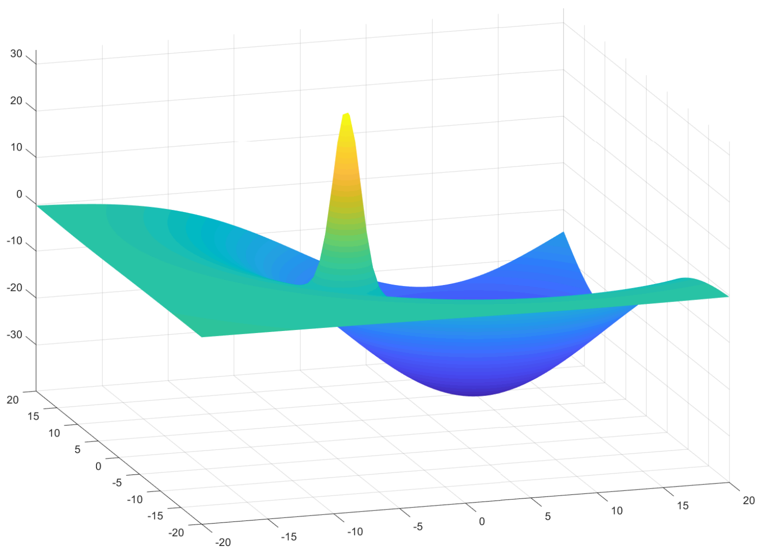

Figure 5.3. Map of the Environment with one obstacle and a target

5.3 Reliability of the FLOP control in multi-agent dynamics

For its structure, the FLOP control requires to the dynamical systems to be in the affine form $\dot{\boldsymbol{x}} = \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{B}\boldsymbol{u}$ and to the cost function to be in the form $E(\boldsymbol{x}, \boldsymbol{u}) = g(\boldsymbol{x}) + \boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u}$. What here is important to highlight is:
- The FLOP formulation does not need any change to be applied on a swarm, i.e. an high number of agents;
- The cost function can provide individual tasks for every agent;
- The cost function can also provide a response to the eventual interactions between agents;
- The cost function can provide collective tasks for the entire swarm;
- The cost function can provide a response to the interactions between agents and the environment.

Starting with the first point in the list, it is straightforward that if the state vector of the system is composed by the state vector of every agent, the FLOP formulation can be no affected from the rise of the number of drones: starting from the statement equation

$$\min \bar{J} = \int_0^T g(\boldsymbol{x}) + \boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u} + \boldsymbol{\lambda}^T(\dot{\boldsymbol{x}} - (\boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{B}\boldsymbol{u}))dt \tag{5.8}$$

it results that if $\boldsymbol{x} = [x_1; x_2; \ldots; x_N]$, the FLOP control theory remains perfectly applicable to the new problem. Therefore, there is no need of any change for applying the FLOP control method to a swarm of dynamical systems instead of a single one. Here a one dimensional state vector is chosen, but this does not lead into any loss of generality. As in the basic formulation, the $g(x)$ function can be applied on a single agent. For instance, let suppose that the swarm is here composed just from $N = 2$ agents, from which: $\boldsymbol{x} = [x_1; x_2]$. In this case, it is possible to describe a function $g(\boldsymbol{x}) = [g(x_1); g(x_2)]$. Regardless from what the $g(x_1)$ and the $g(x_2)$ mean, every agent may have its proper individual task (or more than one individual task). Nevertheless, remaining in the $N = 2$ agents case, it is possible to write a cost function which may account for every agent all other agents, e.g. the only other agent in this case. This can be made by having $g(\boldsymbol{x}) = [g(x_1, x_2); g(x_2, x_1)]$. By writing the proper cost function, is clearly possible to assign collective tasks to the swarm. For example, writing the center of mass of the swarm as x_{CM}, it is possible to write a function whose describe the relationship (e.g. the distance to maintain) from every agent of the swarm and the center of mass, in the form: $g(\boldsymbol{x}) = [g(x_1, x_{CM}); g(x_2, x_{CM})]$. Assigning to the environment some high/low penalty function, it is possible to provide an interaction between the environment and the agents. Let suppose in the environment there is an obstacle, and let assign to the obstacle the state x_{obs}. With the proper cost function (i.e. a repulsive one) it is possible to provide an obstacle avoidance strategy to the agents for the obstacles. Clearly, the quality of the obstacle avoidance will be more precise the most the cost function dedicated will be able to mimic the real obstacle on the environment. In this case, the cost function will be in the form: $g(\boldsymbol{x}) = [g(x_1, x_{obs}); g(x_2, x_{obs})]$. The identical approach can be used in a multi-obstacle environment, without losing of generality or precision. In fact, it is sufficient to write $\boldsymbol{x_{obs}} = [x_{obs_1}; x_{obs_2}; \ldots; x_{obs_{N_{obs}}}]$, with N_{obs} the number of obstacles to provide an obstacle avoidance strategy for a multi-obstacles environment. The objective of the section is to provide an example in the field of swarm robotics, using simple omnidirectional robots, whose each dynamical system is in the form: $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}$, where: $\boldsymbol{x} = [x; y; \dot{x}, \dot{y}]$ and $\boldsymbol{u} = [0; 0; u_x; u_y]$. The idea is to show the possibility given by the usage of a nonlinear $g(\boldsymbol{x})$ in the cost function. In this example, individual and collective tasks are assigned to the agents of the swarm:

- Target reaching: all agents must move to a precise point of the space indicated with $\boldsymbol{x_T}$, the target point.
- Obstacle avoidance: every agent must avoid the obstacles on the path, which are depicted in figure 5.4
- Internal avoidance: each *i-th* quadcopter must avoid all other j quadcopters.
- Formation keeping: agents must maintain an assigned formation, i.e. in this case they must rotate around the center of mass of the swarm with a given radius, proportional to the obstacle's one.

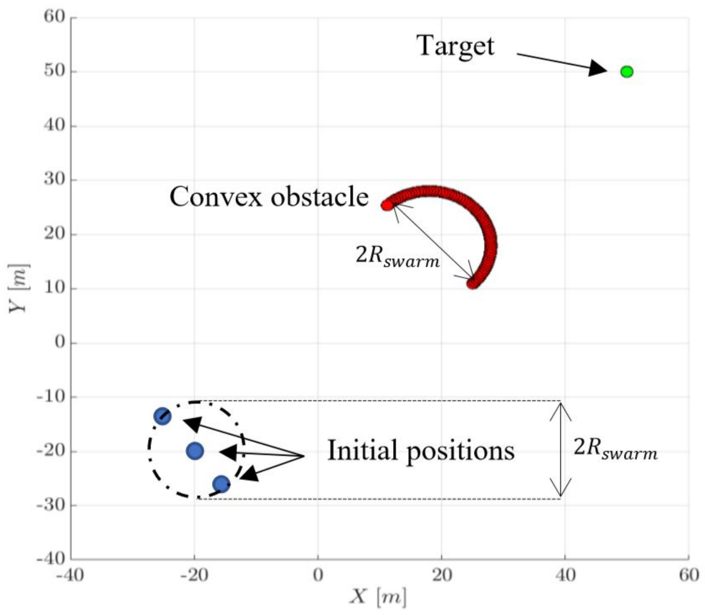

Figure 5.4. Map of the Environment with convex obstacle

The choice of the kind of formation is not random. Looking at the environment, it is straightforward that without a 'collective strategy' it is impossible to reach the target, since the convexity of the obstacle. All agents will reach the internal area of the convex obstacle and they will not succeed in completing the mission of reaching the assigned target. By asking to the agent to rotate around the center of mass with a given radius, it is probable that an high number of robots will succeed in reaching the target. First, the cost funtion is written and it is divided in four parts:

1. **Go to target**: the first task for each agent is to go to the target location, so a quadratic cost function is here introduced:

$$g_T(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{x_T})^T \boldsymbol{Q} (\boldsymbol{x} - \boldsymbol{x_T}) \tag{5.9}$$

   where the velocities in the target vector are set to zero.

2. **Obstacle Avoidance**: the obstacle avoidance is managed with a gaussian-like function, so that the influence of the obstacle is localized close to the obstacle position:

$$g_{OA}(\boldsymbol{x}) = \frac{K_{OA}}{\sqrt{2\pi|\boldsymbol{\Sigma_{OA}}|}} \sum_{i=1}^{N} \sum_{j=1}^{N_{obs}} \exp\left[(\boldsymbol{x_i} - \boldsymbol{x_j})^T \boldsymbol{\Sigma_{OA}}^{-1} (\boldsymbol{x_i} - \boldsymbol{x_j})\right] \tag{5.10}$$

   where K_{OA} is a tuning parameter which make the gaussian-like function become higher (and so the repulsion between the N agents of the swarm), $\boldsymbol{\Sigma_{OA}}$ is the covariance matrix, which represents the correlation between agents.

3. **Internal Avoidance**: similarly to the previous one, the internal avoidance is performed with a gaussian-like function, so that:

$$g_{IA}(\boldsymbol{x}) = \frac{K_{IA}}{\sqrt{2\pi|\boldsymbol{\Sigma_{IA}}|}} \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \exp\left[(\boldsymbol{x_i} - \boldsymbol{x_j})^T \boldsymbol{\Sigma_{IA}}^{-1} (\boldsymbol{x_i} - \boldsymbol{x_j})\right] \tag{5.11}$$

4. **Formation Keeping**: each agent must try to follow an assigned moving position around the center of mass of the swarm. Since every agent try to reach the target, the center of mass is moving and so does the assigned position for each agent.

$$g_F(\boldsymbol{x}) = \frac{K_F}{\sqrt{2\pi|\boldsymbol{\Sigma_F}|}} \sum_{i=1}^{N} \exp\left[(\boldsymbol{x_i} - \boldsymbol{x_{F_i}})^T \boldsymbol{\Sigma_F}^{-1} (\boldsymbol{x_i} - \boldsymbol{x_{F_i}})\right] \tag{5.12}$$

The complete cost function is written as the sum of the four cost function written above, so that:

$$g(\boldsymbol{x}) = g_T(\boldsymbol{x}) + g_{OA}(\boldsymbol{x}) + g_{IA}(\boldsymbol{x}) + g_F(\boldsymbol{x}) \tag{5.13}$$

To test the effectiveness of the method, a probabilistic study is performed, with the aim of analyzing the percentage of the drones of the swarm reaching the target, when increasing the number of individuals of the swarm. The key performance index is the success probability

$$P_{succ} = \frac{N_{Tgt}}{N} \tag{5.14}$$

where N_{Tgt} indicates the number of agents of the swarm which succeeds in reaching the target, while N is the total number of agents. For each number of agents, $n = 30$ simulations were performed with different initial conditions.
As expected, for low number of agents the probability success for the mission is very low. In the case of single agent and two agents, the success probability is equals to zero. This was expected because no rules for avoiding convex obstacle was

implemented in the FLOP method through the g function. In figure 5.5 it can be seen that, by increasing the number of agents, the number of successes increases with a satisfactory rate.

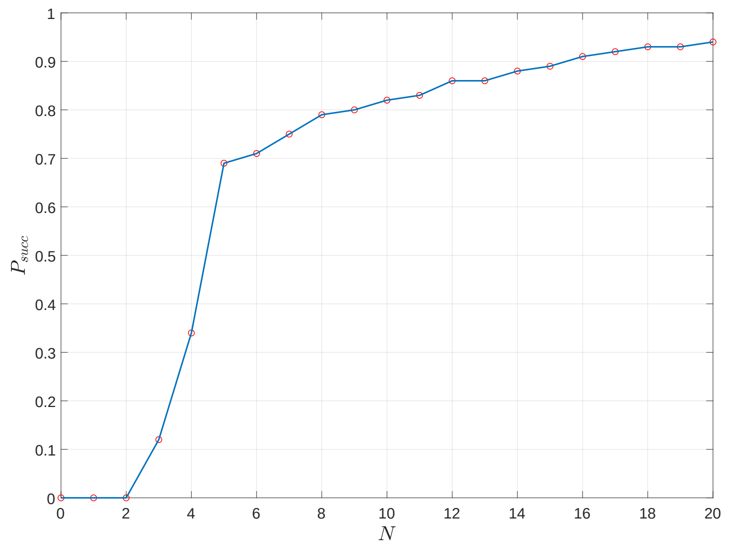

Figure 5.5. Success probability of the mission with increasing number of agents

The interesting fact of this first example related to the swarm robotics relies in the possibility of obtaining a collective behaviour, or intelligence, giving to the swarm only local rules. In fact, each one of the four given tasks is localized and individual. It is localized in the sense that every agent has no complete knowledge of all the environment, but it has a partial knowledge. How partial is the knowledge depends on the variance/covariance $\mathbf{\Sigma}$ matrices in the cost function. The cost function is also individual, since every agent 'think for itself', there is no request for the entire swarm, and each agent is individually controlled.
Here, some points are highlighted due their importance for this discussion:

- the FLOP method proved to be useful in the control of multi-agent dynamics;
- the FLOP method is able to create *ad hoc* functions able to create different tasks for any controlled agent;
- the g function, due the chance to be highly nonlinear, it can create localized penalty functions, which provide interesting results as: obstacle avoidance, internal crash avoidance, rotation around a given point;
- the chance to work with multi-agent systems permit to have interesting result with 'simple' single agents, creating a collective intelligence of the swarm.

In particular, the last point resides exactly in the definition of Dorigo [69]: *"Swarm robotics is the study of how a large number of relatively simple physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment"* For this

reasons, in next sections some example with more complex multi-agent dynamics are investigated:
- quadcopters
- unicycles
- marine drones

5.3.1 The quadcopters dispersion

An Unmanned Aerial Vehicle (UAV) is an aircraft without direct operator manipulation. Many different types of UAVs are nowadays studied but among many, the quadcopters are maybe the most used in today engineerings. A quadcopter [94, 95] is a type of aerial vehicle which uses four rotors equally spaced for its actuation. The four rotors are used to move the quadcopter in three degrees of freedom, three translations and three rotations, and they are considerable the inputs of the problem. In literature there are many studies regarding the control of a quadcopter through different methods such as MPC, LQR, PID controls [45, 96]. In this application, a swarm of quadcopters [95], controlled with the FLOP method, must avoid a fixed obstacle and create a dispersion around the center of mass of the swarm which must be placed on the target position. The intent of this application is to show the ability of the FLOP technique in controlling complex and nonlinear dynamical systems, creating not only the possibility of the movimentation of a single quadcopter, but also a collective intelligence. In this case, the concept of dispersion is highlighted: all quadcopters move from one area of the environment to another, having different tasks:

- Target reaching: all quadcopters has to move to a precise point of the space, indicated with $\boldsymbol{x_T}$, the target point.
- Obstacle avoidance: each quadcopter must avoid the obstacle on the path, which in this case is a solid column placed in $\boldsymbol{x_{Obs}}$.
- Internal avoidance: each *i-th* quadcopter must avoid all other j quadcopters.
- Dispersion: since the target point is the same for all quadcopters of the swarm, the goal is to create a dispersion around the target position. According to the previous section, it is requested to the robot to place itself as far as possible from all other robots, but also to maintain the communication active.

System dynamics and cost function

The position of the quadcopter is defined in the Earth inertial frame x, y, z by the position vector $\boldsymbol{\xi} = [x; y; z]$. The trim is defined using the Euler angles $\boldsymbol{\eta} = [\phi; \theta; \psi]$, which represent the pitch, roll and yaw angle respectively. The system is written using the convention $\boldsymbol{q} = [\boldsymbol{\xi}; \boldsymbol{\eta}]$. The velocity components in the mobile frame are: $\boldsymbol{V} = [u; v; w]$ for the linear velocity and $\boldsymbol{\nu} = [p; q; r]$, with $\boldsymbol{\zeta} = [\boldsymbol{V}; \boldsymbol{\nu}]$. The rotational matrix from the body reference to the inertial frame is $\boldsymbol{R_{b2I}}$

$$\begin{aligned} \boldsymbol{R_{b2I}} =& \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S\phi - S_\psi C_\phi & C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi C_\theta & s_\psi S_\theta S\phi - C_\psi C_\phi & S_\psi S_\theta C_\phi - C_\psi S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix} \\ \dot{\boldsymbol{\xi}} =& \boldsymbol{RV} \end{aligned} \tag{5.15}$$

while $\boldsymbol{W}$ is the matrix which correlates the Euler angles with the angular velocities, so:

$$\begin{aligned} \boldsymbol{W} =& \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & C_\theta S_\theta \\ 0 & -S_\phi & C_\theta C_\phi \end{bmatrix} \\ \dot{\boldsymbol{\eta}} =& \boldsymbol{W\nu} \end{aligned} \tag{5.16}$$

The equation of motion can be written using the Euler-Lagrange equations:

$$\begin{aligned} \dot{\boldsymbol{q}} =& \boldsymbol{J}(\boldsymbol{\nu})\boldsymbol{\zeta} \\ \boldsymbol{M}\dot{\boldsymbol{\zeta}} + \boldsymbol{C}(\zeta)\boldsymbol{\zeta} =& \boldsymbol{\tau} + \boldsymbol{\tau_{ext}} \end{aligned} \tag{5.17}$$

where $\boldsymbol{C}$ is the Coriolis matrix, $\boldsymbol{\tau} = [0; 0; T; \tau_\phi; \tau_\theta; \tau_\psi]$ is the vector of input forces and torques, with T the total force given by the four actuators and $\tau_\phi; \tau_\theta; \tau_\psi$ the torques on the three principal axis respectively, while $\boldsymbol{\tau_{ext}}$ represents the gravity and aerodynamics forces and torques.
Moreover,

$$\boldsymbol{J}(\boldsymbol{\nu}) = \begin{bmatrix} \boldsymbol{R_{b2I}} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{W} \end{bmatrix} \tag{5.18}$$

and

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{I}_{3\times3}\frac{1}{m} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I} \end{bmatrix}$$

with $\boldsymbol{I}_{3\times3}$ the identity matrix and $\boldsymbol{I}$ the inertial matrix. The dynamical system is written so that the FLOP method can be applied, and the cost function $g(\boldsymbol{x})$ is now described. Reminding the FLOP formulation, the problem has to be written for the form:

$$\min \bar{J} = \int_0^T g(\boldsymbol{x}) + \boldsymbol{u}^T\boldsymbol{Ru} + \boldsymbol{\lambda}^T(\dot{\boldsymbol{x}} - (\boldsymbol{\phi(x)} + \boldsymbol{Bu}))dt \tag{5.19}$$

the augmented form of the Potryagin's formulation becomes:

$$\begin{cases} \nabla_x g - \nabla_{\boldsymbol{x}}\boldsymbol{\phi}^T\boldsymbol{\lambda} - \dot{\boldsymbol{\lambda}} = \boldsymbol{0} \\ \boldsymbol{R}^T\boldsymbol{u} - \boldsymbol{B}^T\boldsymbol{\lambda} = \boldsymbol{0} \\ \dot{\boldsymbol{x}} = \boldsymbol{\phi(x)} + \boldsymbol{Bu} \\ \dot{\boldsymbol{\lambda}} = \boldsymbol{G\lambda} \end{cases} \quad \forall\, t \in [0, T] \tag{5.20}$$

where $\boldsymbol{G} = -\frac{\boldsymbol{I}}{\Delta_T}$. The multidimensional feedback control law becomes:

$$\boldsymbol{u(x)} = \boldsymbol{R}^{-T}\boldsymbol{B}^T(\boldsymbol{G} + \nabla_{\boldsymbol{x}}\boldsymbol{\phi})^{-1}\nabla_{\boldsymbol{x}}g \tag{5.21}$$

The cost funtion is divided in three parts:

1. **Go to target**: the first task for each agent is to go to the target location, so a quadratic cost function is here introduced:

$$g_T(\boldsymbol{q}) = (\boldsymbol{q} - \boldsymbol{q_T})^T \boldsymbol{Q}(\boldsymbol{q} - \boldsymbol{q_T}) \tag{5.22}$$

   where the velocities in the target vector are set to zero.

2. **Obstacle Avoidance**: the obstacle avoidance is managed with a gaussian-like function, so that the influence of the obstacle is localized close to the obstacle position:

$$g_{OA}(\boldsymbol{q}) = \frac{K_{OA}}{\sqrt{2\pi|\boldsymbol{\Sigma_{OA}}|}} \sum_{i=1}^{N} \sum_{j=1}^{N_{obs}} \exp\left[(\boldsymbol{q_i} - \boldsymbol{q_j})^T \boldsymbol{\Sigma_{OA}}^{-1}(\boldsymbol{q_i} - \boldsymbol{q_j})\right] \tag{5.23}$$

   where K_{OA} is a tuning parameter which make the gaussian-like function become higher (and so the repulsion between the N agents of the swarm), $\boldsymbol{\Sigma_{OA}}$ is the covariance matrix, which represents the correlation between agents.

3. **Internal Avoidance**: similarly to the previous one, the internal avoidance is performed with a gaussian-like function, so that:

$$g_{IA}(\boldsymbol{q}) = \frac{K_{IA}}{\sqrt{2\pi|\boldsymbol{\Sigma_{IA}}|}} \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \exp\left[(\boldsymbol{q_i} - \boldsymbol{q_j})^T \boldsymbol{\Sigma_{IA}}^{-1}(\boldsymbol{q_i} - \boldsymbol{q_j})\right] \tag{5.24}$$

The total cost function is written as the sum of the above functions, so that:

$$g(\boldsymbol{q}) = g_T(\boldsymbol{q}) + g_{OA}(\boldsymbol{q}) + g_{IA}(\boldsymbol{q}) \tag{5.25}$$

Here, $N = 10$ quadcopters are placed in random initial positions, as depicted in Figure 5.6

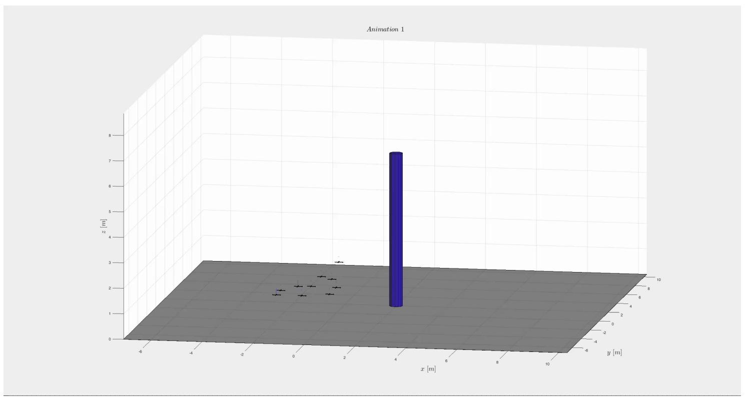

Figure 5.6. Map of the Environment with column-like obstacle

The target point $\boldsymbol{q_T}$ is placed behind the column obstacle, with starting point of view in the quadcopters initial positions. In Figure **??** and **??** a screenshot during the simulation and at the end of it, respectively, show the ability of the FLOP controlled quadcopters to avoid the column-like obstacles, avoid all possible crashes with other agents and to create the 'dispersion' goal around the target position.

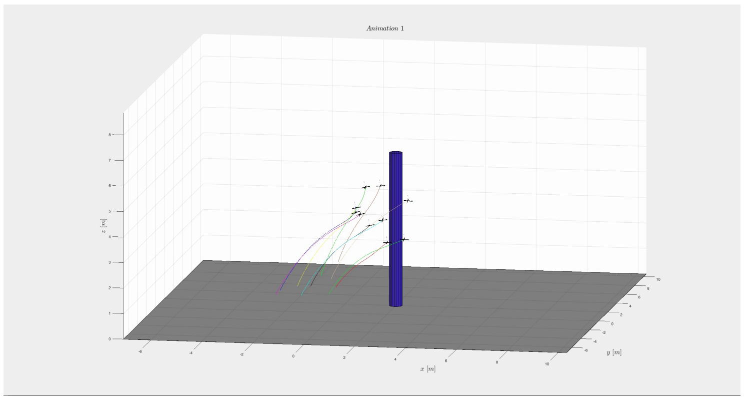

Figure 5.7. Avoiding the Obstacle

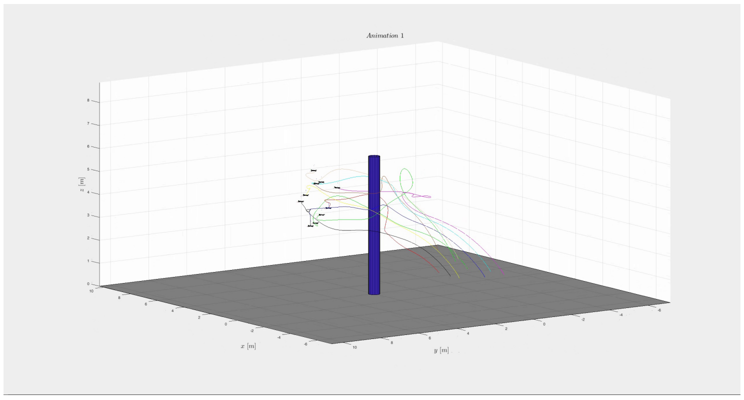

Figure 5.8. Completing the dispersion task

The main points to highlight in this section are:
- the ability of the FLOP method in the control of a multi-quadcopters swarm, and therefore the ability to control highly nonlinear multi-agents systems;
- the reliability of the highly nonlinear cost functions designed for the objective;
- the possibility to extend concepts regardless the complexity of the dynamical system involved.

5.3.2 The migration problem

Among many, the study and analysis of the migration and the collective transport of swarm of robots are of interest. Through the study of stigmergy, it is possible to identify the interaction processes that give rise to intelligent cooperative systems, capable of performing complicated collective operations. The world stigmergy derives from Greek words *stigma* and *ergon*, meaning 'mark or sign' and 'work or action', respectively. Stigmergy [97] represents the universal coordination mechanism: a consensus mechanism of indirect coordination within an environment among agents or actions. In nature, agents follow very simple rules, and even without the need for centralized control, global behavior emerges, unknown to the individual agents, who can find efficient methods of transport and migration. One of the first efficient collective transport is given by the Vicsek particle model [82] which each agent follows a collective group velocity. This model is widely used to imitate the movement of shoals of fish and swarms of birds that manage to move in a coordinated way, following environmental stimuli. Based on this model, many studies have been developed concerning the coordinated collective transport of robots. In particular, the generalization of the Vicsek model [98–100] to robot movement concerns two types of models:

1. a first class does not involve anti-collision rules allowing for collisions between robots

2. uses sophisticated sensors and communication hardware that make the swarm collision-free.

In this application the communication template presents some simplifications with respect to a fully all-to-all connected system, balancing short-range and long-range transmission of information within the swarm. This gives the robot to equip the agents with exteroceptive sensors present on the market in such a way as to analyze the state of the agents around them, implementing the actions provided by the control strategy. Furhtermore, appropriate control logics introduce an effective internal avoidance rule between agents. For this reason, a method of analyzing information from neighbouring agents is proposed, which combines the most significant aspects of the simplified analysis of the first neighbours only and the complete global analysis. The proposed navigation system of a swarm of robots is divided into two main categories: collective exploration and coordinated motion [101]. Here, unicycle robots move in an unknown environment and navigates without internal collisions with other agents, trying to migrate from a start to a target zone using the information provided by the neighbors' agents to reduce the migration time. These different tasks are achieved by using the innovative feedback control Feedback Local Optimality Principle - FLOP - method. The FLOP approach permits to apply simultaneously the collective exploration and the coordinated motion strategies. The environment, represented in figure 5.10, present many obstacles randomly put, where each obstacle represent a hill. The distributed control uses only local velocity informations to drive the agents of the swarm in a small region where the signal velocity is captured. The agents follow a nonlinear control strategy where each of them is tracking a target velocity resulting in a directional averaging operation. This process is a mimick of the behavior of ant colonies in which the information travels with pheromones permitting to move around obstacles of various types together with a high migration speed. The purpose of this example is to show the ability of the FLOP method in the control of a large population of cooperative agents.

System dynamics and cost function

The single agent, represented on the left of Figure 5.9, is intended as a unicycle model [102]and its dynamics is expressed as:

$$\begin{bmatrix} X \\ \dot{Y} \\ \dot{\psi} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \boldsymbol{M}^{-1} \begin{bmatrix} v\cos(\psi) \\ v\sin(\psi) \\ \omega \\ f_T - c_l v - \nabla_X h(x,y)\sin(\psi) + \nabla_Y h(X,Y)\cos(\psi) \\ f_M - c_r\omega \end{bmatrix} \tag{5.26}$$

where $\boldsymbol{M} = \text{diag}[1; 1; 1; m; I]$. The X, Y, ψ, v, ω, m, I, c_l, c_r and $h(X,Y)$ are the spatial coordinates of each agent, the heading orientation, the longitudinal speed, the rotational speed, the mass, the rotational inertia, the two-speed resistance coefficients (longitudinal and rotational) and the potential function representing the unknown environment respectively.

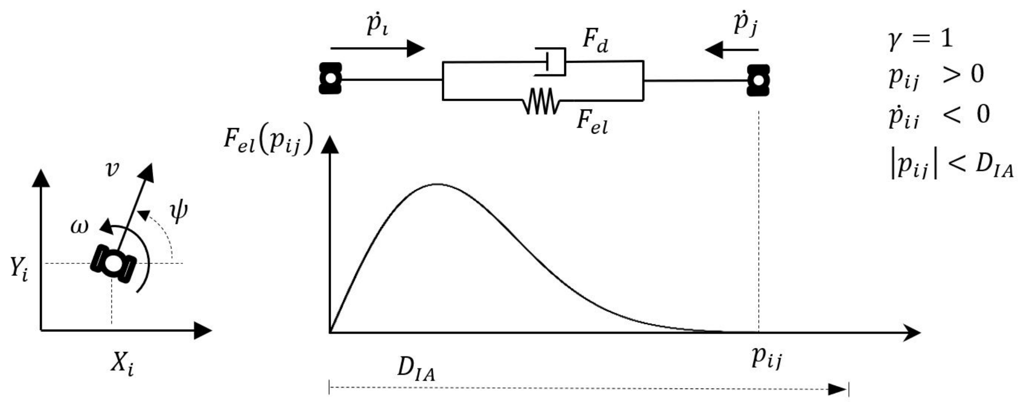

Figure 5.9. Unicycle model and representation of the elastic force

In this example, the robots are controlled by the thrust force f_T and the yaw moment f_M. The $\nabla_X h(X,Y)$, $\nabla_Y h(X,Y)$ represents the external gravity force. The state vector is here defined as $\boldsymbol{x} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_N}]$, while the control vector is $\boldsymbol{u} = [\boldsymbol{u_1}, \boldsymbol{u_2}, \ldots, \boldsymbol{u_N}]$ for the N - robot system, and considering each state vector as: $\boldsymbol{x_i} = [X_i; Y_i; \psi_i; v_i; \omega_i]$ and each control vector as $\boldsymbol{u} = [f_{T_i}; f_{M_i}]$ is simple to organize the full nonlinear system as:

$$\dot{\boldsymbol{x}} = \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{B}\boldsymbol{u} \tag{5.27}$$

The cost function is expressed as

$$E(\boldsymbol{x}, \boldsymbol{u}) = \frac{1}{2}\boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u} + g(\boldsymbol{x}) \tag{5.28}$$

where $g(\boldsymbol{x}) = g_{CE}(\boldsymbol{x}) + g_{CM}(\boldsymbol{x})$ ant those two terms represent the collective exploration (CE) and cooperative motion (CM) task respectively.
The **collective exploration** task regards every single agent: in fact, each agent must migrate to one are of the environment to another, having an assigned target location $\boldsymbol{x_T}$. Moreover, each agent must avoid all ather agents through an internal avoidance rule.
It requires two effects:

- Rendezvous: all agent must reach an assigned location $\boldsymbol{x_T}$. This task is here often referred also as Go to Target:

$$g_{GtT}(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{x_T})\boldsymbol{Q}^T(\boldsymbol{x} - \boldsymbol{x_T}) \tag{5.29}$$

- Internal Avoidance: each agent must not collide with any of the other agents. The internal avoidance penalty function g_{IA} is written as the sum of two terms, one for the relative positions between agents $\boldsymbol{p_{ij}} = \boldsymbol{p_j} - \boldsymbol{p_i}$ and one for their respective velocities, $\dot{\boldsymbol{p}}_{\boldsymbol{ij}} = \dot{\boldsymbol{p}}_{\boldsymbol{j}} - \dot{\boldsymbol{p}}_{\boldsymbol{i}}$, with $\boldsymbol{p_i} = [X_i; Y_i]$.

$$\begin{aligned} g_{IA}(\boldsymbol{x}) &= \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \left(\frac{K_{IA}}{\sqrt{2\pi|\Sigma_{IA}|}} \exp - \left[\boldsymbol{p_{ij}}^T \boldsymbol{\Sigma_{IA}}^{-1} \boldsymbol{p_{ij}} \right] + \gamma \dot{\boldsymbol{p}}_{\boldsymbol{ij}}^T \boldsymbol{Q_{IA}} \dot{\boldsymbol{p}}_{\boldsymbol{ij}} \right) \\ \gamma &= \frac{1}{2} \left(1 + \tanh\Big(- k_{p1}(\boldsymbol{p_{ij}} \cdot \dot{\boldsymbol{p}}_{\boldsymbol{ij}}) \Big) \tanh\Big(- k_{p2}(||\boldsymbol{p_{ij}}|| - D_{IA}) \Big) \right) \end{aligned} \tag{5.30}$$

The first addend of the g_{IA} is a Gaussian function and its gradient is depicted on the right of Figure 5.9, where the repulsive elastic force F_{el} between agents is represented. The Gaussian parameters $\Sigma_{IA}, |\Sigma_{IA}|, K_{IA}$ represents the variance-covariance matrix, its determinant and a gain factor respectively. All their parameters are studied so that the maximum elastic force F_{el} is high enough to avoid any kind of crash between agents, considering their maximum velocities. The second addend is a quadratic potential function of the relative speed $\dot{\boldsymbol{p}}_{\boldsymbol{ij}}$, and its gradient represents a dissipation force F_D that is activated and deactivated as a function of gamma. In particular, the dissipation force is turned on when two agents find themselves at a distance closer than a chosen threshold D_{IA} and they have a relative speed that identifies an imminent collision, given by the sign of the scalar product between the relative velocities and positions of two agents. Tuning parameters k_{p1}, k_{p2} permit to obtain a smoother or higher slope for γ .

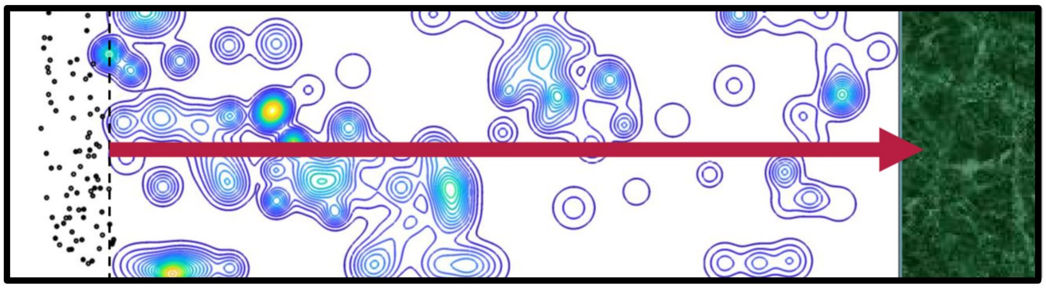

Figure 5.10. Map of the environment

The **coordinated motion** strategy has the aim to increase the performance of the migration of the swarm, by giving to each agent some information on the velocities of its surrounding agents. It introduces an adding term to the cost function, $g_{CM}(\boldsymbol{x})$ which expresses the ability of every single drone to go the area near him with the highest average speed in the direction of the target. The cost function is written as:

$$g_{CM}(\boldsymbol{x}) = \sum_{i=1}^{N} (\boldsymbol{s_i} - \boldsymbol{s_i}^*)^T \boldsymbol{Q_{CM}} (\boldsymbol{s_i} - \boldsymbol{s_i}^*) \tag{5.31}$$

where $\boldsymbol{s_i} = [\psi_i; v_i, \omega_i]$, and s_i^* is determined by the strategy proposed below. The *i-th* agent can observe a portion of the surrounding environment, called $\mathcal{S}_\rangle$, which is assumed as a sector of the circle of radius R_{max}, centered on the agent position. The section is also delimited by the two lines associated with the angles α_0 and α_{End}, measured with respect to the x-axis of the *i-th* agent. $\mathcal{S}_\rangle$ is further divided into N_{Zones} sub-sectors or zones, named Z_k, with $k = 1, \ldots, N_{Zones}$, so that each zone is a sector of angle $\hat{\alpha} = \frac{\alpha_{End} - \alpha_0}{N_{Zones}}$. In $\mathcal{S}_\rangle$, the agent searches for all the agents currently within Z_k, the number of which is denoted by J_k . For each agent j therein, its velocity component in the direction of the target $v_{jk,T}$ is observed. Here,the chosen direction of the target is along the Y direction. If r_{ijk} is the distance of the *j-th* agent within Z_k from the *i-th* observer agent, the weighing numbers are defined:

$$w_{ijk} = -r_{ijk} + R_{max} \tag{5.32}$$

The weighted velocity $V_{ik,T}$ is estimated by the *i-th* observer for the zone Z_k as:

$$V_{ik,T} = \frac{\sum_{j=1}^{J_k} w_{ijk} v_{jk,T}}{\sum_{j=1}^{J_k} w_{ijk}} \tag{5.33}$$

and the highest value within the sector is selected as:

$$V_{i,T} = \max\left(V_{ik,T}\right) \forall\, k \;\in\; [1, N_{Zones}] \tag{5.34}$$

Once the sector with the highest velocity is found, the agent steers its velocity in the direction of the bisector of the zone k^* with the highest velocity. In Figure 5.14 the coordinated motion strategy is represented.

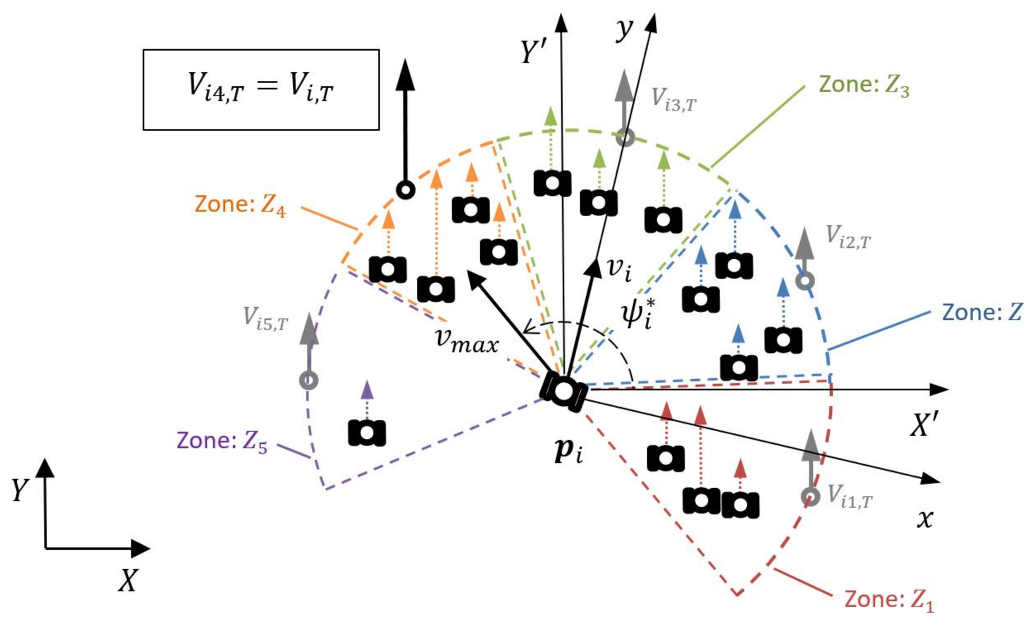

Figure 5.11. Coordinated Motion strategy representation for a single agent

The target is positioned along the vertical axes. In the case depicted in Figure 5.14, the highest weighted velocity is in the zone Z_4 so $k_i^* = 4$. The angle of the desired maximum velocity v_{max} is $\psi^* = \alpha_0 + \left(k_i^* - \frac{1}{2}\right)\hat{\alpha}$, so $\boldsymbol{s}_i^*$ in equation (5.31) becomes:

$$\boldsymbol{s}_i^* = [\psi_i^*; v_{max}; 0] \tag{5.35}$$

Simulation and Results

The discussed strategy is now tested on a complex environment, depicted in Figure 5.14. As it can be seen, all agents are initially placed in a starting area and must complete the migration to another area of the environment. Each agent is controlled by the FLOP method, and two different strategies are tested:
- in the first strategy, the collective exploration is active.
- in the second strategy, the coordinated motion is added to the collective exploration strategy.

The number of agents of the swarm increases from $N = 5$ to $N = 40$, with $\Delta N = 5$. For each chosen number of robots, $n_s = 60$ simulations are performed, and the arrival time is recorded. The arrival time here is intended as the time of arrival of the last agent. The arrival time for both the first and the second strategy is shown in Figure 5.12.

Time.pdf

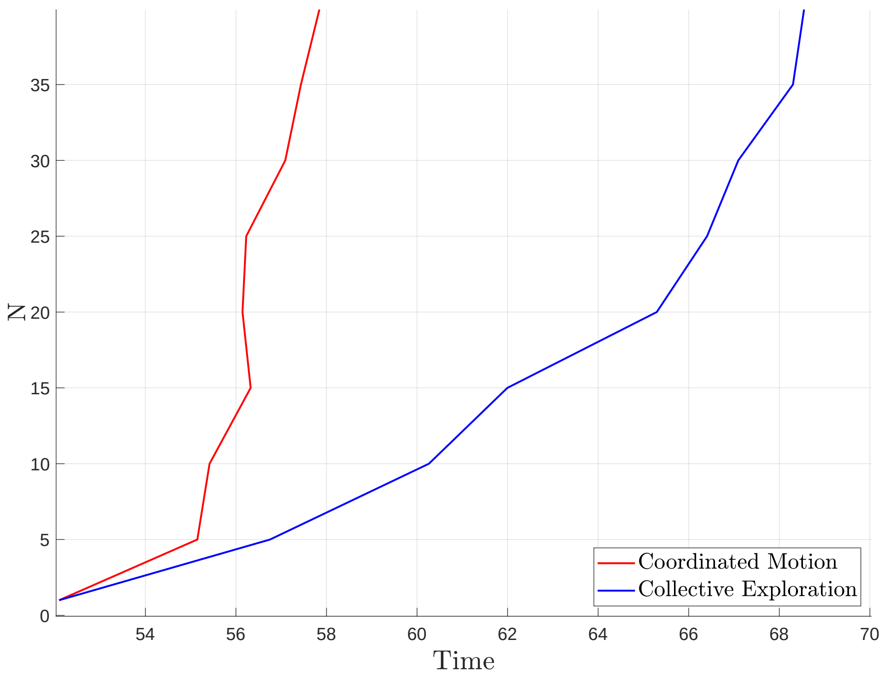

Figure 5.12. Arrival time for the last agent in different swarms

Moreover, a Probability Density Function (PDF) of the arrival time for $N = 85$ agents is shown in Figure 5.13.

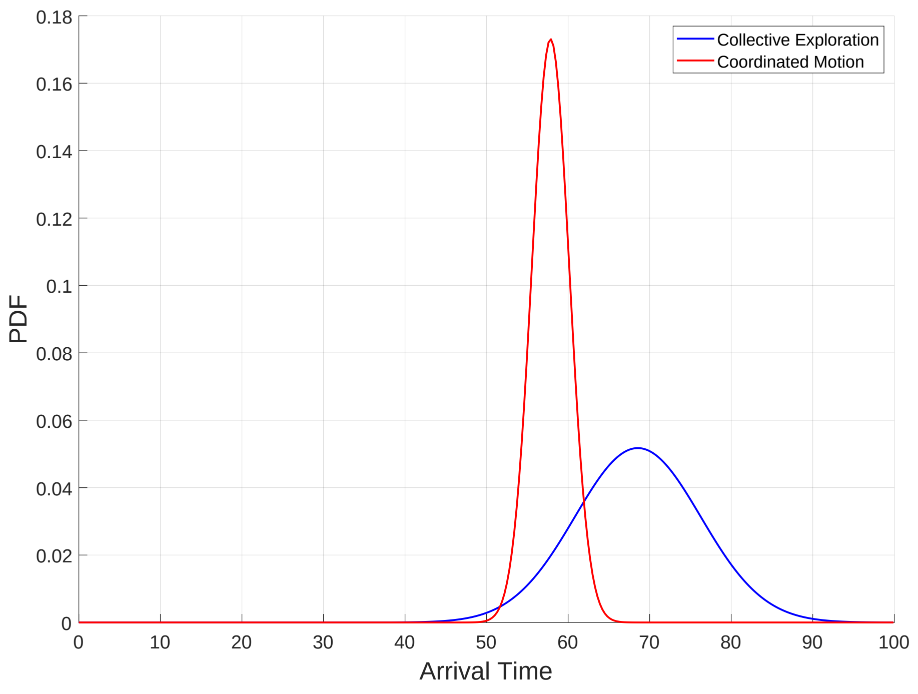

Figure 5.13. Probability Density Function of the arrival time for $N = 85$ agents

As it is possible to see, the coordinated motion strategy succeeds in being more efficient in terms of arrival time than the collective exploration one. The more are the agents, the more the gain in terms of final arrival time is higher. This is proven in Figure 5.13, with the PDF for the $N = 85$ agents swarm. By varying the initial conditions for $n_s = 60$ simulations, results are very promising for the coordinated motion strategy. In Figure 5.14, screenshots at different times give an idea of the organization of the swarm that the coordinated motion strategy can provide. There is a stigmergic movement of the swarm, which is able to develop two 'preferred lines' to perform the migration.

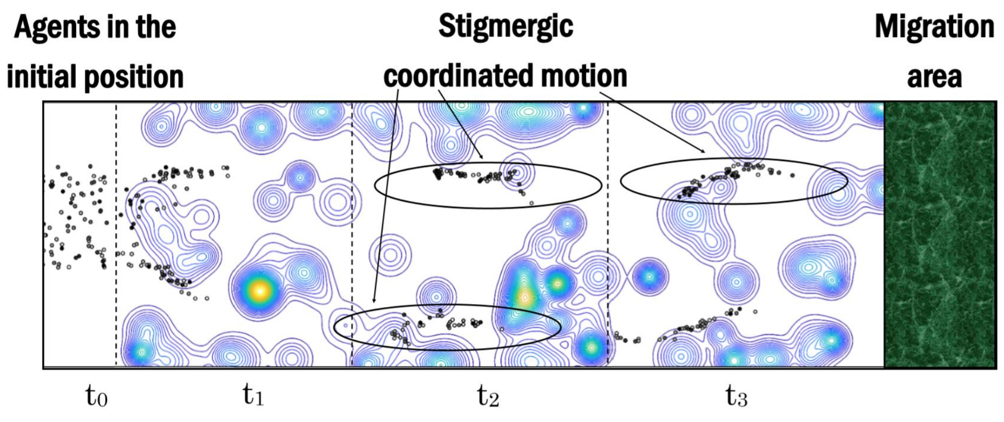

Figure 5.14. Stigmergic movement of the swarm

Comment to the main results In this application the FLOP proved to be a very useful control in swarm robotics applications. A swarm of a number up to 85 unicycle robots has been successfully controlled by the Feedback Local Optimality Principle. Here, the main task for each robot was to avoid all collisions with other robots, while performing a migration in a complex environment, i.e. an environment full of obstacles. Two different strategies were performed: in the first one every agent knows the position of the target area and is able to recognize if there is any close agent in order to avoid collisions. Hence, this first strategy is an individual strategy, in which each agent try to complete its individual mission, i.e. to migrate without collisions. In the second strategy a collective intelligence is created, through a strategy here named coordinated motion. Each agent has a knowledge of the velocity fields of closest agents, and adjusts its velocity by orienting it in the direction of the area in which the average velocity in the target direction is higher.
The coordinated motion strategy succeeds in creating a stigmergic movement of the swarm, which with sufficiently high number of agents create an important advantage in terms of arrival time of the last agent, i.e. the time in which the migration is completed.
The FLOP method proved to be useful in three different ways:

1. The agents are all identified with a nonlinear dyamical system, which the FLOP succeeds in controlling it.

2. The collective exploration strategy includes a non-quadratic cost function for the internal obstacle avoidance, so that if two agents are closer than a chosen threshold and they are moving one towards the other, a highly nonlinear, localized cost function is placed on the colliding agent and the collision can be avoided.

3. The FLOP can compute the solution of a complex problem with an high number of robots with low computational costs. In fact, in each simulation the computational time was much lower than the evaluated time of the simulation, despite the high number of robots used.

5.3.3 The breakdown boat rescue

Sea drones have the potential to revolutionize maritime business, reducing time, cost and risk of several operations. The recent extraordinary evolution of the robotic technology, together with the increase of computational capabilities and the use of intelligent sensors, has driven the intense implementation of robot swarms, i.e. cooperative and coordinated sets of intelligent agents able to achieve common goals. One of the major challenges in this research field is related to the identification of intelligent control logics capable of managing a large number of robots working jointly for a common purpose. The control systems, acting on every single agent, are meant for a collaborative configuration. This duality is the key to achieve high flexibility and the power to face and manage unpredictable events. Here, the FLOP algorithm is applied to autonomous driving boats whose task is the transport of a damaged or in distress boat by encircling it. Different research aspects are studied and developed such as control approaches (of both the single agent and the

entire swarm formation), communication among the robots, coordination strategies, learning methodologies, transportation problems. The employment of swarm based strategies has effectively proven the reliability and robustness towards failures and unpredictable events during operations; a resilience evaluation for swarm algorithms is reported in [103]. This latter is of particular interest in the scope of autonomous goods and material handling, where in the near future the appearance of autonomous robotic couriers is likely to happen [104].
Another potential remarkable application for cooperative robotic swarms deals with Search and Rescue activities. Particularly, if in other domains (ground, air) such frameworks are more advanced and robotics swarms result in more effective operations, the marine environment poses bigger challenges, in terms of both sensing capabilities and control performance. In general, many surveys and overviews conducted on existing rescue robotics systems underline how there is much further work to be carried on; in particular, work in [105] highlights how robotics systems employed in real disasters are still too tied to human supervision, as well as they usually consists of single-robots working alone. Specifically dealing with search and rescue in marine environments, a focus on the potentiality of autonomous underwater vehicles is provided in [106], pointing out that there is much work still to be performed in order to become effective and timely in actual applications. A more recent experience described in [107] confirms the need of further development and technology consolidation. Because of the benefits provided, a number of studies have been steered toward the development and employment of swarm-based approaches for the guidance and control of fleets of autonomous platforms in marine and maritime contexts. The work [108] presented the integration of a swarm aggregation scheme with a path-following guidance module for USVs (Unmanned Surface Vehicles): the swarm algorithm allowed the vehicles to maintain a fixed-range formation, while the guidance system drove the entire team along a desired path. The work has been further extended including a path-planner module allowing the employment of the swarm formation within harbor contexts [109]. A number of researches have been oriented towards the problem of providing support to ships, e.g. tug operations for berthing aid. The work [110] presents a strategy that allows a swarm of autonomous tugboats to cooperatively move a large object on the water, keeping into account the actuator limitations and complex hydrodynamics; however it only examines some different tugboat configurations without considering the approach strategy. Furthermore, in the latter work, the vehicles are identical and also the distressed agent is part of the robotic swarm (making the towing operation more difficult); finally, in [110] a PID control is employed. A similar problem is faced in [111] where a team of autonomous vessels is commanded to perform a cooperative towing operation employing an adaptive position controller. A complementary result is provided by the work [112], where a nonlinear observer is designed in order to estimate the state of a vessel towed by multiple autonomous boats; a sliding mode controller is further developed in order to guide the motion of the vessel by means of the autonomous towing platforms.
The problem of removing or re-positioning of a vessel in distress through a swarm of marine drones is considered. It is not unusual that a vessel needs to be assisted during marine operations. One of the major reasons of the assistance is failure, still it is not the only one. One of the possible examples is related to sail boats that

have to enter the port. Their actuation is generally not able to perform a suitable maneuver within the port restricted space, and hence they are usually supported by one or more rubber boats. Indeed, particularly within very restricted spaces, a vessel may need to be turned or moved from a point to another one, due to lack in its maneuvering capabilities, e.g. because its actuation system is not suitable for that kind of motion. For these reasons, given the interest in the application, the paper presents an approach for the rescue maneuvering of a distressed vessel, exploiting a swarm of marine drones. The distressed vessel is assumed in a total breakdown, as the worst case is here addressed.
The proposed paper improves the exploitation of the swarm, with respect to the previously cited works, by introducing both individual approach strategies, i.e. internal avoidance and target reaching, and a smart aggregation scheme for the collective transportation of the distressed vessel. The final goal for the swarm is to transfer the distressed vessel from the breakdown location to an assigned target area. The rescuing swarm is built in new generation smart and soft materials, thus making the "pushing actions" possible without causing damages to the robotic structure itself.
Given the novelty of the proposed approach, only the methodological and theoretical aspects are dealt with at this stage, neglecting technological issues such as sensor modeling, communication infrastructures and non-destructive bumping. These issues will be faced as soon as a real-case framework will be set up for practical testing of the approach.

Single agent dynamics and cost function

Here, the dynamical system used for the single agent of the swarm is described as a planar three DoF's model [113]. Ignoring heave (vertical), pitch and roll motion, the dynamics is expressed as:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\psi} \\ \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} = \boldsymbol{M}^{-1} \begin{bmatrix} u\cos\psi - v\sin\psi \\ v\cos\psi + u\sin\psi \\ r \\ rv + F_{drag_u} \\ -ru + F_{drag_v} \\ F_{drag_r} \end{bmatrix} + \boldsymbol{M}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ u_1 + u_2 \\ u_3 \\ \frac{Lu_3 + u_1 w - u_2 w}{4} \end{bmatrix} + \boldsymbol{M}^{-1} \left[\boldsymbol{\tau_{i2BD}} + \boldsymbol{\tau_{i2j}}\right] \tag{5.36}$$

where $\boldsymbol{M} = diag\,[1;1;1;m;m;I]$. The variables X, Y, ψ, u, v, r, m, I represent the coordinates in the $2D$ fixed reference, the heading orientation, the longitudinal and lateral speed, the rotational speed, the mass and the rotational inertia of the single agent, respectively. The symbols F_{drag_u}, F_{drag_v}, F_{drag_r} are drag resistances in the mobile reference frame. Each agent has two aft propellers and one lateral propeller, which produce u_1, u_2 and u_3 respectively. All boats are here supposed identical, with L, w, h to be the length, width and height of each boat respectively. Forces and torques $\boldsymbol{\tau_{i2BD}}$, $\boldsymbol{\tau_{i2j}}$ are between the $i-th$ agent of the swarm and the breakdown boat and between the $i-th$ and $j-th$ agents of the swarm, respectively. Both $\boldsymbol{\tau_{i2BD}}$ and $\boldsymbol{\tau_{i2j}}$ are intended of two types: crash forces and frictional forces. Fixed and mobile reference system are placed as depicted in Figure **??**. In Figure 5.15

an emulation of the pushing phase is shown. In general, $\boldsymbol{\tau_:}$ is intended as a six rows vector written as: $\boldsymbol{\tau_:} = [0; 0; 0; \boldsymbol{\tau_{El}} + \boldsymbol{\tau_{Fric}}]$, where $\boldsymbol{\tau_{El}}$ and $\boldsymbol{\tau_{Fric}}$ represents the elastic and friction forces and torques along the longitudinal, lateral and height direction. Elastic force is evaluated as:

$$\boldsymbol{\tau_{El_{1:2}}} = K_{el} A \boldsymbol{\hat{n}} \tag{5.37}$$

where K_{el} depends on the soft material of which the agents are made, A is the contact area of which the centroid is individuated by point C, $\hat{n}$ is orthogonal to the two contact points H_1, H_2, which individuates the direction of the tangential versor $\hat{\tau}$. Friction force is evaluated as:

$$\boldsymbol{\tau_{Fric_{1:2}}} = \mu_d ||(\boldsymbol{\tau_C})|| sign(\boldsymbol{V_r}) \boldsymbol{\hat{\tau}} \tag{5.38}$$

where μ_d is a friction coefficient of the chosen material, $||(\tau_C)||$ is the Euler-norm of τ_C and $sign(\boldsymbol{V_r})$is the sign of the relative velocity between the two colliding agents. Both torques due the elastic and friction forces are evaluated by using as a position vector the vector between the centroid of the boat and the centroid C of the contact area.

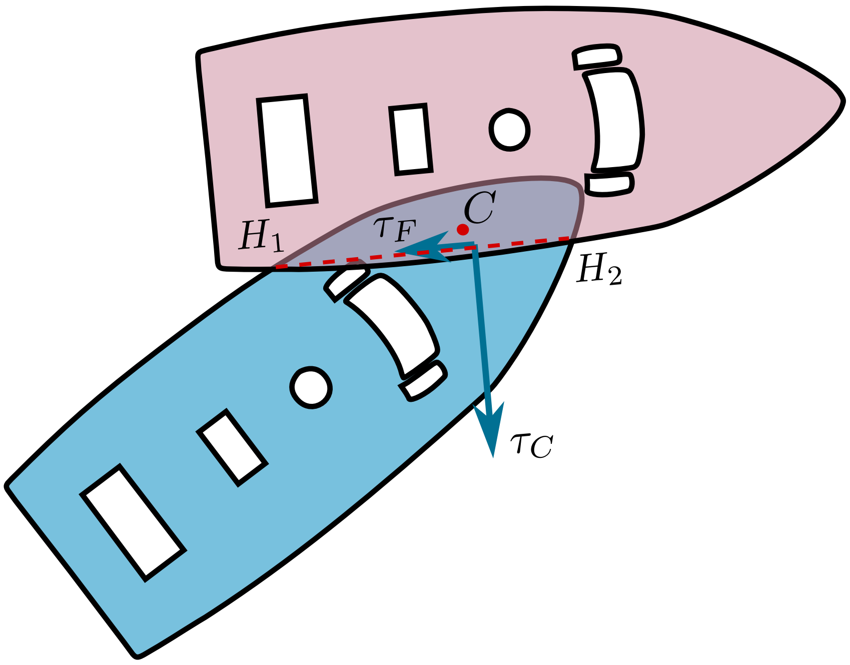

Figure 5.15. Push and friction forces

it is possible to write the state and control vector of the $i-th$ agent as $\boldsymbol{x_i} = [X_i,\ Y_i\ \psi_i, u_i,\ v_i,\ r_i]$ and $\boldsymbol{u_i} = [u_{i1},\ u_{i2},\ u_{i3}]$, respectively. In the case of N agents in the swarm, state and control vector are defined as $\boldsymbol{x} = [\boldsymbol{x_1},\ \boldsymbol{x_2},\ \ldots,\ \boldsymbol{x_N},\ ]$ and $\boldsymbol{u} = [\boldsymbol{u_1}, \boldsymbol{u_2},\ \ldots, \boldsymbol{u_N}]$, so that the full nonlinear dynamic system can be written in the affine form:

$$\boldsymbol{\dot{x}} = \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{Bu} \tag{5.39}$$

The breakdown boat is intended as one of the agent of the swarm, but fully uncontrolled, so that the worst case scenario for rescuing is depicted. Its dynamical system is the same of the controlled boats in (5.36), with $\boldsymbol{u} = \mathbf{0}$, $\boldsymbol{\tau_{i2j}} = \mathbf{0}$. The breakdown boat and the rescue (target) state vector are introduced as $\boldsymbol{x_{BD}}$ and $\boldsymbol{x_{Tgt}}$.
Different strategies for the towing assignment are discussed. Since the nature of the forces taken into account, the agents can only push the breakdown boat in the direction of the target, but they are not able to pull the vessel.
Therefore, the environment is divided into a 'active area' and a 'non-active area'. First, the two areas are identified. Then, three strategies for towing the breakdown boat are discussed.
The first one is the Individual Rescue Strategy or IRS, in which each agent must push the breakdown boat in the target direction.
The second one is the Collective Rescue Strategy (CRS), in which a pushing formation is individuated.
The third one is the Swarm Rescue Strategy (SRS), in which the entire swarm try to put the breakdown boat into the center of mass of the swarm and then to move as a unique group to the target area. It is reasonable to expect that the IRS should not be very effective, but it is here used as benchmark for the other strategies. The CRS would be very effective from a theoretical point of view, but from a practical point of view it would require to know a lot of *a priori* information, for example the size of the agents, their distance from the target, the number of agents that are coming etc.
The SRS should work adaptively, regardless of the number of drones (as long as $N > 1$). It is reasonable to think that it could suffer from orientation problems, but it is equally reasonable that, with a fairly large number of drones, it can work ensuring the recovery of the damaged boat.
Inizialization of the rescue The problem starts by assigning random initial positions to all agents, the breakdown boat and the target position. First, the environment is divided into two different areas:

- Active area
- Non-active area

The non-active area is an arc of a circle of angle $\beta_{NA} = 2\hat{\beta}$, centered on the angle between the breakdown boat and the target position ψ_{BD2Tgt}. This area, depicted in Figure 5.16, represents the zone in which agents can not push the breakdown boat to the target. If any robot find itself in this area at any point of the simulation, it is forced to go at a chosen checkpoint location $\boldsymbol{\xi_{CP}}$ evaluated as follows:

$$\boldsymbol{\xi_{CP}} = \begin{bmatrix} X_{BD} \\ Y_{BD} \end{bmatrix} + R\left(\psi_{BD2Tgt} + \pi\right) \begin{bmatrix} -k_{CP}L \\ 0 \end{bmatrix} \tag{5.40}$$

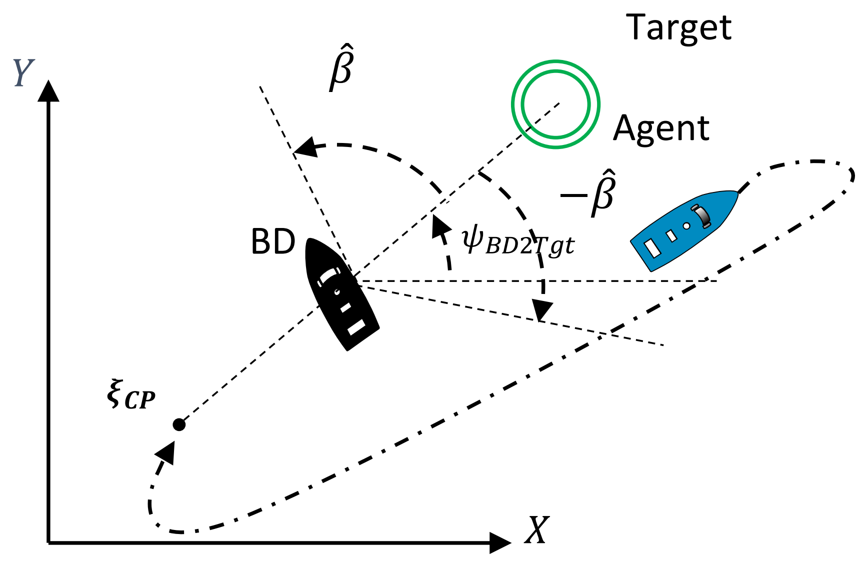

Figure 5.16. Study case environment

Anytime an agent is in the towing area, must try to rescue the breakdown boat. To this aim, three different towing strategies are illustrated:

- the *Individual Rescue Strategy* (IRS)
- the *Collective Rescue Strategy* (CRS)
- the *Swarm Rescue Strategy* (SRS)

In the IRS each agent has the individual task to reach the breakdown boat and to push it by orienting itself in the direction of the target. In the CRS, it is asked to all agents to keep a chosen formation and to pursuit the individual task given in the IRS. In the SRS, all agents must reach the breakdown boat (IRS) and simultaneously they must go to the center of mass of the swarm. When the mean distance of all agents from the center of mass is lower than a chosen threshold, the center of mass is attracted to the target position. In each strategy, two different phases can be described:

- The approach phase
- The pushing phase

Individual Rescue Strategy

In the approach phase, each agent try to reach a determined approach point $\boldsymbol{\xi_A} = [X_A,\ Y_A]$ near the breakdown boat. The approach point is determined starting from the coordinates in the fixed reference of the breakdown boat, using the angle ψ_{BD2Tgt} in the fixed reference between the breakdown boat and the target, so that

$$\boldsymbol{\xi_A} = \begin{bmatrix} X_{BD} \\ Y_{BD} \end{bmatrix} + R\left(\psi_{BD2Tgt} + \pi\right) \begin{bmatrix} -k_A L \\ 0 \end{bmatrix} \tag{5.41}$$

where $k_A \in (0,\ 1)$ and the following notation is assumed: $R(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}$. Example of the placement of approach point $\boldsymbol{\xi_A}$ is illustrated in Figure 5.17.

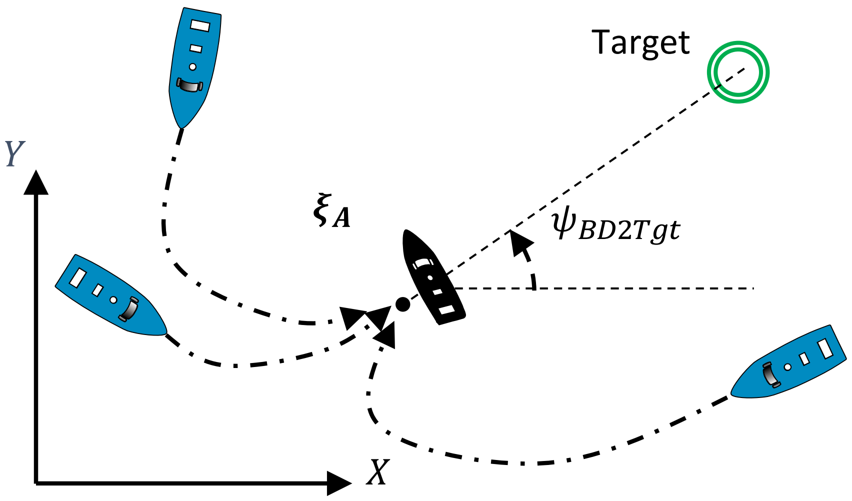

Figure 5.17. Approach phase for IRS

The cost function related to the approach phase of each agent $g_A(\boldsymbol{x_i})$ is represented as it follows:

$$g_A(\boldsymbol{x_i}) = (\boldsymbol{x_i} - \boldsymbol{x_A})^T \boldsymbol{Q}_A (\boldsymbol{x_i} - \boldsymbol{x_A}) \tag{5.42}$$

where $\boldsymbol{x_A} = [\boldsymbol{\xi_A};\ \psi_{BD2Tgt};\ 0,\ 0;\ 0]$ and $\boldsymbol{Q_A}$ is an appropriate gain matrix. Following phase differs from strategy to strategy. The first approach is intended as an individual try of each agent to pursue the final objective to move the vessel in distress to the desired target. Therefore, each agent has the individual objective to push the breakdown boat in the direction of the target. To perform this purpose, each agent receives two tasks:

- to orient its heading in the direction of the target
- to impose $k_A = 0$.

This can be written as a general cost function for the individual rescue strategy $g_{IRS}(\boldsymbol{x_i})$, so that it is the sum of the approach phase cost function $g_A(\boldsymbol{x_i})$ and the pushing phase's one, $\boldsymbol{g}_{i2Tgt}(\boldsymbol{x_i})$.

$$\begin{aligned} g_{IRS(\boldsymbol{x_i})} &= \boldsymbol{g}_A(\boldsymbol{x_i}) + \boldsymbol{g}_{i2Tgt}(\boldsymbol{x_i}) = \\ &= (\boldsymbol{x_i} - \boldsymbol{x_A})^T \boldsymbol{Q}_A (\boldsymbol{x_i} - \boldsymbol{x_A}) + \\ &+ (\boldsymbol{x_i} - \boldsymbol{x_{Tgt}})^T \boldsymbol{Q}_{2Tgt} (\boldsymbol{x_i} - \boldsymbol{x_{Tgt}}) \end{aligned} \tag{5.43}$$

where $\boldsymbol{x_{Tgt}} = [0;\ 0;\ \psi_{i2Tgt};\ 0,\ 0;\ 0]$ and $\boldsymbol{Q_{2Tgt}}$ is an appropriate gain matrix.

Collective Rescue Strategy

Here, each drone takes into account also the position of other agents and tries to establish a given position in a designed formation. The required formation is designed with similar idea respect to the one for the design of the approach phase of the IRS. The approach point for each drone $\boldsymbol{\xi_{iA}}$ is written as:

$$\boldsymbol{\xi_{iA}} = \begin{bmatrix} X_{BD} \\ Y_{BD} \end{bmatrix} + R\left(\psi_{BD2Tgt} + \beta_i\right)\begin{bmatrix} -k_A L \\ 0 \end{bmatrix} \tag{5.44}$$

where β_i is designed so that: $\sum_{i=1}^{N} \beta_i = 0$ and $\beta_i \leq \beta_{max}$, with β_{max} previously decided. Example of of the approach phase for CRS is illustrated in Figure 5.18.

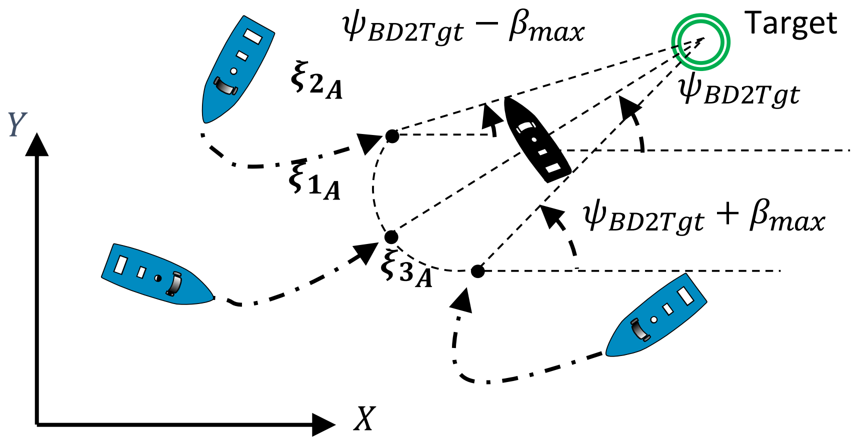

Figure 5.18. Approach phase for CRS

In a similar way to (5.42), the cost function for the approach phase in the CRS is:

$$g_{A_i}\left(\boldsymbol{x_i}\right) = \left(\boldsymbol{x_i} - \boldsymbol{x_{i_A}}\right)^T \boldsymbol{Q}_A \left(\boldsymbol{x_i} - \boldsymbol{x_{i_A}}\right) \tag{5.45}$$

where this time $\boldsymbol{x_{i_A}} = [\boldsymbol{\xi_{i_A}};\ \psi_{BD2Tgt} + \beta_i;\ 0,\ 0;\ 0]$ so that each agent has an individual approach point. The pushing phase in the CRS follows the same methodology of the IRS case, so the cost function of the CRS can be directly written as:

$$\begin{aligned} g_{CRS}\left(\boldsymbol{x_i}\right) &= g_{A_i}\left(\boldsymbol{x_i}\right) + \boldsymbol{g}_{i2Tgt}\left(\boldsymbol{x_i}\right) = \\ &= \left(\boldsymbol{x_i} - \boldsymbol{x_{i_A}}\right)^T \boldsymbol{Q}_A \left(\boldsymbol{x_i} - \boldsymbol{x_{i_A}}\right) + \\ &+ \left(\boldsymbol{x_i} - \boldsymbol{x_{Tgt}}\right)^T \boldsymbol{Q}_{2Tgt} \left(\boldsymbol{x_i} - \boldsymbol{x_{Tgt}}\right) \end{aligned} \tag{5.46}$$

Swarm Rescue Strategy

In the last strategy each agent has both individual and collective tasks, which can be summarized as it follows:

- approach phase (as in the IRS);
- minimization of the distance from itself and the center of mass of the swarm;
- following the center of mass which is asked to go in the direction of the target;

The first task is taken from the IRS without any difference. As supplementary request, to each agent is asked to minimize the distance between itself and the center of mass of the swarm. When the variance of all the distances between agents and the center of mass is lower than a given threshold, it is asked to the center of mass to reach the desired target. This last requirement represents the pushing phase for the SRS. First, the center of mass of the swarm is evaluated as

$$\boldsymbol{\xi_{CM}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\xi_i} \tag{5.47}$$

where $\boldsymbol{\xi_i} = [X_i;\ Y_i]$, and a vector for the center of mass is defined as: $\boldsymbol{x_{CM}} = [\boldsymbol{\xi_{CM}};\ \mathbf{0};\ \mathbf{0};\ \mathbf{0};\ \mathbf{0};\ Y_i]$. A quadratic cost function for the attraction to the center of mass $g_{CM}\left(\boldsymbol{x_i}\right)$ is evaluated as:

$$g_{CM}\left(\boldsymbol{x_i}\right) = \left(\boldsymbol{x_i} - \boldsymbol{x_{CM}}\right)^T \boldsymbol{Q}_{CM}\left(\boldsymbol{x_i} - \boldsymbol{x_{CM}}\right) \tag{5.48}$$

The variance of relative distances between agents and the center of mass is:

$$\sigma = \sum_{i=1}^{N}\sqrt{\frac{\left(X_i - X_{CM}\right)^2 + \left(Y_i - Y_{CM}\right)^2}{N}} \tag{5.49}$$

The cost function for the movement of the center of mass in the direction of the target is

$$g_{2Tgt}\left(\sigma\right) = \left(\boldsymbol{x_{CM}} - \boldsymbol{x_{Tgt}}\right)^T \boldsymbol{Q}\left(\sigma\right)\left(\boldsymbol{x_{CM}} - \boldsymbol{x_{Tgt}}\right) \tag{5.50}$$

where $\boldsymbol{Q}\left(\sigma\right) = \left[q_1\left(\sigma\right);\ q_2\left(\sigma\right);\ 0;\ ;\ 0;\ 0\right]$ and $q\left(\sigma\right) = \frac{\hat{q}}{\sigma}$. All considered, the SRS cost function is written as:

$$g_{SRS}\left(\boldsymbol{x_i}\right) = g_{IRS}\left(\boldsymbol{x_i}\right) + g_{CM}\left(\boldsymbol{x_i}\right) + g_{2Tgt}\left(\sigma\right) \tag{5.51}$$

Simulation and Results

irst results for IRS, CRS and SRS strategies are illustrated. For all strategies, the study case scenario is as it follows: the breakdown boat is positioned at a distance of around 300 meters from the target position, with random initial heading. The swarm boats are placed in random positions in a 'box' around the breakdown boat of approximately 100 meters, with random heading as well. All vessels are identical. For each random initial conditions, all three strategies are performed and around 100 simulations with different initial conditions are performed. Since the equal dimensions of the breakdown boat and the agents, maximum number of agents considered is $N = 5$. Each simulation is considered successful if the breakdown boat arrives in proximity of the target, with a circular threshold with radius is L . In Fig. 5.19 the success rate of each strategy in the performed simulations is shown, while in Fig. 5.20 the probability density function of the arrival time at the target with $N = 4$ is depicted. As it can be seen, CRS and SRS are not performed for $N = 1$, because they required at least $N = 2$ agents. All three strategies gave promising results, with a minimum success probability around the 80% in correspondence of the maximum number of agents. As it was expected, the SRS had best results in terms of success probability, but as it can be seen in , it is slower both than the CRS and the IRS. However, the IRS is quicker of the other two strategies, but it is the one with the lower success probability. This was addressed to the lack of control in the heading variable during the SRS strategy, which is very robust in term of results but slower than the CRS.

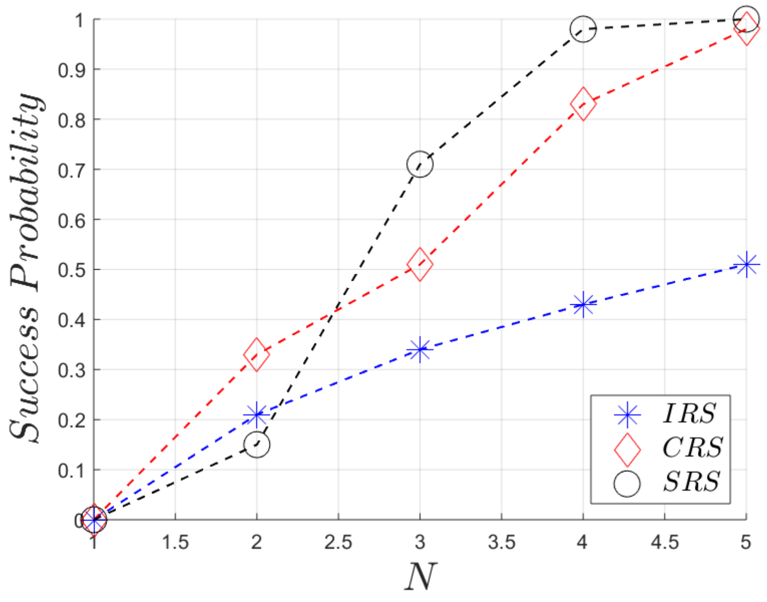

Figure 5.19. Success Probability

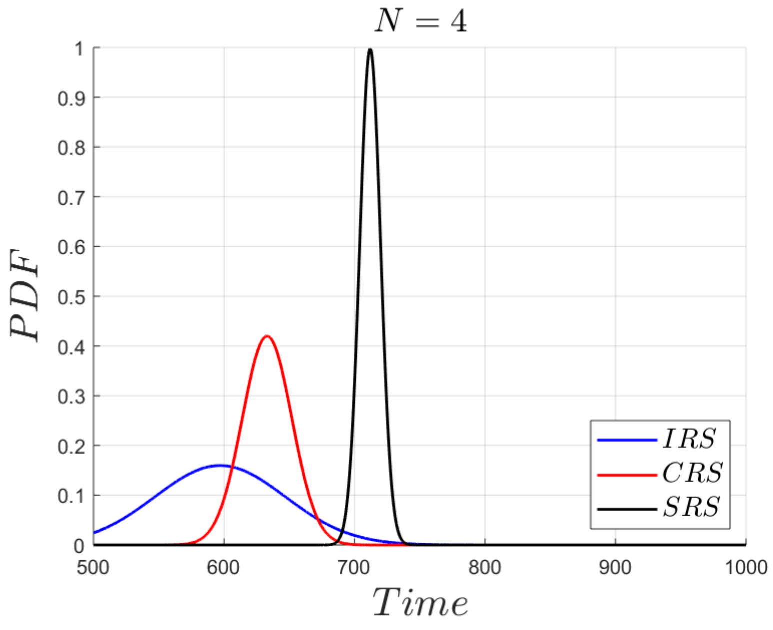

Figure 5.20. Probability Density Function for $N = 4$

Comment to the main results

The FLOP method provides the possibility to control each agent of the swarm with individual task, and to give a collective task to the entire swarm. Different towing strategies were illustrated, and some preliminary results for each strategy has been given. Although the initial state of the project, FLOP method succeeds in rescuing the breakdown boat with all three tested strategies. Further developments will comprehend a more complex dynamics for both agents and the breakdown boat, different sizes and shapes for each agent. The chance to have different size between agents and the breakdown boat will give the possibility to use larger numbers of agents for the swarm. Nevertheless, showed results give a first insight of the capability of the FLOP control to manage the cooperation between agents with different tasks assigned.

Chapter 6

Conclusions

The goal of this work was to define and present a new formulation for a feedback control named Feedback Local Optimality Principle - FLOP - and to show some of its potential applications in the field of mechanical engineering.
The problem of control is one of the biggest challenges in modern engineering. The first two chapters introduce the problem of control, from classical techniques to the most modern ones. Particular attention was given to the Optimal Control Problems theory, which works using the calculus of variation, one powerful and useful tool in the field of control. The goal of Optimal Control Problems theory is to find the optimal trajectories both in the input variable, i.e. the control and in the dependent variable, i.e. the state of the system. When this happens, the global minimum of a performance index, known as 'functional' is found. The functional is a scalar value which is described by an integral from the initial to the final time of the evolution of the system, and it is composed by a cost function which needed to be minimized. The integral is subjected to some constraints, i.e. the differential equation whose represents the dynamic evolution of the system and the initial condition, usually assumed known.
From an engineering point of view, Optimal Control Problems presents some issues which need to be overcome:

1. the optimal trajectories may be very difficult (and even impossible) to find analytically;
2. if the optimal trajectories are found, they lead to a feed-forward control law, which leads to (i) the impossibility to account any error on the modeling of the process, (ii) the impossibility to account any random disturbance during the evolution of the system;
3. the optimal trajectories can require a high computational cost to be found.

The proposed FLOP method aims to overcome the difficulties of the Optimal Control Problems by proposing a new algorithm able to provide a feedback control law, starting from the basics principles of the Optimal Control Problems and using the power of the calculus of variation as basics for the method.
The new control algorithm succeeds in giving a control law to a sufficiently general class of dynamical systems and therefore can be applied to different relevant

engineering problems. After a mathematical derivation produced in chapter 3, in chapters 4 and 5, the FLOP algorithm has been tested in prototypal problems and in the field of Swarm Robotics.
Swarm Robotics is one of the most intriguing areas in modern engineering, due to the galore of possible applications and the convergence of technologies, from math to physics and to electrical and mechanical engineering. The applicability of the FLOP control in the field of Swarm Robotics highlight its robustness, reliability, and flexibility. As test cases, in the previous chapter the FLOP control results able to control indifferently swarm of quadcopters, unicycles, and marine drones, in three different environments and with different individual and collective tasks. On the one hand, the FLOP control proved to be able to control the dynamics of single agents, from simple to complex one. On the other hand, it proved to be useful also in creating some collective behaviors with nonlinear dynamics, which sounds very promising for further developments.
A huge part of modern engineering moves in the direction of autonomous vehicles, whether they are Aerial, Ground or Marine drones. The problem of autonomous navigation is huge and regards many aspects of engineering, from sensors to actuators and the control logic. In this last field, many studies arose in last decades and many different techniques were implemented starting from more classical theories. The complex the dynamical system to be controlled, the complex the control logic is needed. The control logic here proposed is very flexible and can be used in all categories of unmanned vehicles: in this work all three categories are investigated. The possibilities for the FLOP control are wide and in this work some of them are investigated. Its capabilities are limited only by the imagination of the users since its possibilities are very wide. For instance, in related works, the FLOP control was also tested on the control of an autonomous car for steering control, couple with obstacle avoidance innovative techniques. The FLOP method has to be intended as a tool for controlling dynamical systems, able to give in result sub-optimal (local optimal) trajectories. Clearly, it is not aimed to be compared to global optimization techniques such as the Optimal Control Problem's ones, while it is aimed to be compared to whose techniques such as Model Predictive Controls, the Linear Quadratic Regulators or Direct and Indirect Methods for Optimal Control. Comparing to these techniques, the main disadvantage belongs in the mathematical request of a special class of dynamical systems and cost functions, while the biggest advantage relies on the low computational costs and the simplicity of the mathematical formulation.
These considerations are the starting points of the future developments of this work, which firstly concern the follow-up of the experimental campaign. Besides, since the aim is to present the more general feedback control possible, some of the hypothesis made for the formulation of the control should be removed.

Bibliography

[1] B. Brian Park and Seongah Hong. "Traffic Flow Stabilization Strategy for Mitigating Automated and Human Driven Vehicles Interactions". In: *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2018. DOI: 10.1109/iecon.2018.8591075.

[2] Mujahid Abdulrahim. "Flight Dynamics and Control of an Aircraft with Segmented Control Surfaces". In: *42nd AIAA Aerospace Sciences Meeting and Exhibit*. American Institute of Aeronautics and Astronautics, 2004. DOI: 10.2514/6.2004-128.

[3] Enzo Kopperger et al. "A self-assembled nanoscale robotic arm controlled by electric fields". In: *Science* 359.6373 (2018), pp. 296–301. DOI: 10.1126/science.aao4284.

[4] Omer Morgul. *On the strain feedback control of a flexible robot arm*. 2011. DOI: 10.1109/acc.2011.5991063.

[5] Jun Zhou and Yueqing Yu. "Coordination control of dual-arm modular robot based on position feedback using Optotrak3020". In: *Industrial Robot* 38 (2011), pp. 172–185. DOI: 10.1108/01439911111106381.

[6] H. Bolandi and S. M. Esmaeilzadeh. *Adaptive Nonlinear Sensor Output Feedback Control of a Flexible Robot Arm*. 2008. DOI: 10.1109/iccee.2008.122.

[7] Florian Petit and Alin Albu-Schaffer. *State feedback damping control for a multi DOF variable stiffness robot arm*. 2011. DOI: 10.1109/icra.2011.5980207.

[8] Boris S Kerner. *Introduction to modern traffic flow theory and control: the long road to three-phase traffic theory*. Springer Science & Business Media, 2009.

[9] Swaroop Darbha and KR Rajagopal. "Intelligent cruise control systems and traffic flow stability". In: *Transportation Research Part C: Emerging Technologies* 7.6 (1999), pp. 329–352.

[10] Steven E Shladover, Dongyan Su, and Xiao-Yun Lu. "Impacts of cooperative adaptive cruise control on freeway traffic flow". In: *Transportation Research Record* 2324.1 (2012), pp. 63–70.

[11] Dimitris Bertsimas and Sarah Stock Patterson. "The traffic flow management rerouting problem in air traffic control: A dynamic network flow approach". In: *Transportation Science* 34.3 (2000), pp. 239–255.

[12] Karl LaFleur et al. “Quadcopter control in three-dimensional space using a noninvasive motor imagery-based brain–computer interface”. In: *Journal of Neural Engineering* 10.4 (2013), p. 046003. DOI: 10.1088/1741-2560/10/4/046003.

[13] Yujung Yoon, Mingu Kim, and Youdan Kim. “Three Dimensional Path Planning for Aerial Refueling Between One Tanker and Multiple UAVs”. In: *International Journal of Aeronautical and Space Sciences* 19.4 (Dec. 1, 2018), p. 1027. DOI: 10.1007/s42405-018-0098-z. URL: http://dx.doi.org/10.1007/s42405-018-0098-z.

[14] Jinglan Li, Qinmin Yang, and Youxian Sun. “Robust State and Output Feedback Control of Launched MAVs with Unknown Varying External Loads”. In: *Journal of Intelligent & Robotic Systems* 92.3-4 (Dec. 1, 2018), p. 671. DOI: 10.1007/s10846-018-0774-z. URL: http://dx.doi.org/10.1007/s10846-018-0774-z.

[15] Chunrong Wang, Jing Zhao, and Erdong Xia. “Multi-objective optimal design of a novel multi-function rescue attachment based on improved NSGA-II”. In: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 40.7 (June 13, 2018), p. 1. ISSN: 1806-3691. DOI: 10.1007/s40430-018-1263-9. URL: http://dx.doi.org/10.1007/s40430-018-1263-9.

[16] Wendong Gai et al. “A New Closed-loop Control Allocation Method with Application to Direct Force Control”. In: *International Journal of Control, Automation and Systems* 16.3 (June 1, 2018), p. 1355. ISSN: 2005-4092. DOI: 10.1007/s12555-017-0294-9. URL: http://dx.doi.org/10.1007/s12555-017-0294-9.

[17] Stuart Bennett. “Nicholas Minorsky and the automatic steering of ships”. In: *IEEE Control Systems Magazine* 4.4 (1984), pp. 10–15.

[18] Martin H. Weik. “Nyquist theorem”. In: *Computer Science and Communications Dictionary*. Springer US, 2000, pp. 1127–1127. DOI: 10.1007/1-4020-0613-6_12654.

[19] K.J. Åström and T. Hägglund. “Automatic tuning of simple regulators with specifications on phase and amplitude margins”. In: *Automatica* 20.5 (1984), pp. 645–651. ISSN: 0005-1098. DOI: https://doi.org/10.1016/0005-1098(84)90014-1. URL: http://www.sciencedirect.com/science/article/pii/0005109884900141.

[20] S.P. Bhattacharyya, A. Datta, and L.H. Keel. *Linear Control Theory: Structure, Robustness, and Optimization*. Automation and control engineering. CRC Press, 2009. ISBN: 9781315221182. URL: https://books.google.it/books?id=ur5htgEACAAJ.

[21] Walter R. Evans. “Control System Synthesis by Root Locus Method”. In: *Transactions of the American Institute of Electrical Engineers* 69.1 (1950), pp. 66–69. DOI: 10.1109/t-aiee.1950.5060121.

[22] Y.W. Lee, T.P. Cheatham, and J.B. Wiesner. “Application of Correlation Analysis to the Detection of Periodic Signals in Noise”. In: *Proceedings of the IRE* 38.10 (1950), pp. 1165–1171. DOI: 10.1109/jrproc.1950.233423.

[23] D.E. Kirk. *Optimal Control Theory: An Introduction.* Dover Books on Electrical Engineering Series. Dover Publications, 2004. ISBN: 9780486434841. URL: `https://books.google.it/books?id=fCh2SAtWIdwC`.

[24] John E. Prussing. *Optimal Control Theory.* 2018. DOI: `10.1093/oso/9780198811084.003.0004`.

[25] Velimir Jurdjevic. *Optimal control theory.* DOI: `10.1017/cbo9780511530036.009`.

[26] Bowon Kim. *Optimal Control Theory and Operations Strategy.* 2017. DOI: `10.1007/978-981-10-3599-9_1`.

[27] Larry W. Mays. *Systems Theory and Optimal Control.* 2018. DOI: `10.1201/9781351075206-1`.

[28] Suresh P. Sethi. *What Is Optimal Control Theory?* 2019. DOI: `10.1007/978-3-319-98237-3_1`.

[29] Manoj Kushwah and Ashis Patra. "PID Controller Tuning using Ziegler-Nichols Method for Speed Control of DC Motor". In: *IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012).* 2014.

[30] J.L. Guzmán et al. "Understanding PID design through interactive tools". In: *IFAC Proceedings Volumes* 47.3 (2014). 19th IFAC World Congress, pp. 12243–12248. ISSN: 1474-6670. DOI: `https://doi.org/10.3182/20140824-6-ZA-1003.01328`. URL: `http://www.sciencedirect.com/science/article/pii/S1474667016435639`.

[31] Igor Boiko. *Non-parametric Tuning of PID Controllers.* 2013. DOI: `10.1007/978-1-4471-4465-6_2`.

[32] Jing-Chung Shen. "New tuning method for PID controller". In: *ISA Transactions* 41.4 (2002), pp. 473–484. ISSN: 0019-0578. DOI: `https://doi.org/10.1016/S0019-0578(07)60103-7`. URL: `http://www.sciencedirect.com/science/article/pii/S0019057807601037`.

[33] Man-Fat Yeung. *Automatic design of fuzzy PID controllers.* DOI: `10.14711/thesis-b679890`.

[34] J.D. Simon and S.K. Mitter. "A theory of modal control". In: *Information and Control* 13.4 (1968), pp. 316–353. DOI: `10.1016/s0019-9958(68)90834-6`.

[35] Jean-François Magni. *Modal Control: A Tutorial.* 2002. DOI: `10.1007/978-1-4615-0637-9_3`.

[36] Yann Le Gorrec and Jean-Francois Magni. *Multi-Variable Modal Control.* DOI: `10.1002/9780470612521.ch14`.

[37] Y. H. Chen and E. D. Piontek. *Robust Modal Control of Distributed-Parameter Systems with Uncertainty.* 1990. DOI: `10.23919/acc.1990.4791080`.

[38] Clarence W. deSilva. *Techniques and Applications of Experimental Modal Analysis in Modeling and Control.* 1984. DOI: `10.23919/acc.1984.4788517`.

[39] Victor Korotkich. “Pontryagin Maximum Principle”. In: *Encyclopedia of Optimization*. Springer US, 2008, pp. 2987–2990. DOI: 10.1007/978-0-387-74759-0_512.

[40] L.S. Pontryagin et al. “The mathematical theory of optimal processes (Selected works)”. In: *Classics of Soviet Mathematics* 4 (1986), pp. xxiv+360.

[41] R. Bellman. “Dynamic Programming”. In: *Science* 153.3731 (1966), pp. 34–37. DOI: 10.1126/science.153.3731.34.

[42] L. E. O. N. COOPER and M. A. R. Y. W. COOPER. *One-dimensional Dynamic Programming: Computational Solutions*. 1981. DOI: 10.1016/b978-0-08-025065-6.50009-7.

[43] Sven Danø. *Applications of Dynamic Programming*. 1975. DOI: 10.1007/978-3-7091-8394-6_6.

[44] John O. S. Kennedy. *Introduction to Dynamic Programming*. 1986. DOI: 10.1007/978-94-009-4191-5_2.

[45] Lucas M. Argentim et al. *PID, LQR and LQR-PID on a quadcopter platform*. 2013. DOI: 10.1109/iciev.2013.6572698.

[46] Enrique Barbieri. *On the LQR Problem for Phase-Canonic Systems*. 1993. DOI: 10.23919/acc.1993.4793492.

[47] Xie Huan and Duan Yu. *Research on LQR optimal control method of active engine mount*. 2018. DOI: 10.1063/1.5033639.

[48] Tayfun Çimen. “State-Dependent Riccati Equation (SDRE) Control: A Survey”. In: 41 (2008), pp. 3761–3775. ISSN: 1474-6670. DOI: 10.3182/20080706-5-kr-1001.00635.

[49] S. Sivasundaram. *Nonlinear Problems in Aviation and Aerospace*. 2000. DOI: 10.1201/9781482296914.

[50] Martine Olivi. “The Laplace Transform in Control Theory”. In: vol. 327. Oct. 2006, pp. 193–209. DOI: 10.1007/11601609_12.

[51] Leigh. *Control Theory: A guided tour*. 2012. DOI: 10.1049/pbce072e.

[52] Naval Education. *Gunner’s Mate Missile M 3 And 2*. Periscope Film, LLC, 2013. ISBN: 9781937684327. URL: https://books.google.it/books?id=ahUdngEACAAJ.

[53] B.C. Chachuat. *Nonlinear and Dynamic Optimization: From Theory to Practice - IC-32: Spring Term 2009*. Polycopiés de l’EPFL. EPFL, 2009. URL: https://books.google.it/books?id=%5C_JOHYgEACAAJ.

[54] R. E. Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: 82 (1960), p. 35. ISSN: 0021-9223. DOI: 10.1115/1.3662552.

[55] R. E. Kalman and R. S. Bucy. “New Results in Linear Filtering and Prediction Theory”. In: 83 (1961), p. 95. ISSN: 0021-9223. DOI: 10.1115/1.3658902.

[56] Kruna Ratkovic. “Limitations in direct and indirect methods for solving optimal control problems in growth theory”. In: *Industrija* 44.4 (2016), pp. 19–46. DOI: 10.5937/industrija44-10874.

[57] Paweł Drag et al. "A Review on the Direct and Indirect Methods for Solving Optimal Control Problems with Differential Algebraic Constraints". In: *Recent Advances in Computational Optimization*. Springer International Publishing, July 2015, pp. 91–105. DOI: 10.1007/978-3-319-21133-6_6.

[58] Thomas Carraro and Michael Geiger. *Direct and Indirect Multiple Shooting for Parabolic Optimal Control Problems*. 2015. DOI: 10.1007/978-3-319-23321-5_2.

[59] Dennis Janka, Stefan Körkel, and Hans Georg Bock. *Direct Multiple Shooting for Nonlinear Optimum Experimental Design*. 2015. DOI: 10.1007/978-3-319-23321-5_4.

[60] Christian Kirches. *The Direct Multiple Shooting Method for Optimal Control*. 2011. DOI: 10.1007/978-3-8348-8202-8_2.

[61] Rien Quirynen, Milan Vukov, and Moritz Diehl. *Multiple Shooting in a Microsecond*. 2015. DOI: 10.1007/978-3-319-23321-5_7.

[62] P. W. Meyer. "A fixed point multiple shooting method". In: *Computing* 40.1 (Mar. 1988), pp. 75–83. ISSN: 1436-5057. DOI: 10.1007/BF02242191. URL: https://doi.org/10.1007/BF02242191.

[63] E.F. Camacho and C.B. Alba. *Model Predictive Control*. Advanced Textbooks in Control and Signal Processing. Springer London, 2013. ISBN: 9780857293985. URL: https://books.google.it/books?id=tXZDAAAAQBAJ.

[64] L. Magni, D.M. Raimondo, and F. Allgöwer. *Nonlinear Model Predictive Control: Towards New Challenging Applications*. Lecture Notes in Control and Information Sciences. Springer Berlin Heidelberg, 2009. ISBN: 9783642010941. URL: https://books.google.it/books?id=jxluCQAAQBAJ.

[65] Heiko Hamann. *Swarm Robotics: A Formal Approach*. Springer International Publishing, 2018. DOI: 10.1007/978-3-319-74528-2.

[66] Heiko Hamann. *Introduction to Swarm Robotics*. Springer, 2018, pp. 1–32. DOI: 10.1007/978-3-319-74528-2_1.

[67] Gerardo Beni. *From Swarm Intelligence to Swarm Robotics*. 2005. DOI: 10.1007/978-3-540-30552-1_1.

[68] Gerardo Beni. *Swarm Intelligence*. 2009. DOI: 10.1007/978-0-387-30440-3_530.

[69] M. Dorigo. *SWARM-BOT: an experiment in swarm robotics*. DOI: 10.1109/sis.2005.1501622.

[70] Manuele Brambilla et al. "Swarm Robotics: A Review from the Swarm Engineering Perspective". In: *Swarm Intelligence* 7 (Mar. 2013), pp. 1–41. DOI: 10.1007/s11721-012-0075-2.

[71] Carl Anderson. *Swarm Intelligence: From Natural to Artificial Systems. Eric Bonabeau , Marco Dorigo , Guy Theraulaz*. Vol. 76. Oxford University Press„ Inc., 2001, pp. 268–269. DOI: 10.1086/393972.

[72] C. Blum and D. Merkle. *Swarm Intelligence: Introduction and Applications*. Natural Computing Series. Springer Berlin Heidelberg, 2008. ISBN: 9783540740896. URL: https://books.google.it/books?id=6Ky4bVPCXqMC.

[73] Heiko Hamann et al. “Analysis of emergent symmetry breaking in collective decision making”. In: *Neural Computing and Applications* 21.2 (Apr. 2010), pp. 207–218. DOI: 10.1007/s00521-010-0368-6.

[74] M. Ludwig L. Gini. “Distributed autonomous robotic systems”. In: ed. by Springer. Berlin: Springer, 2006. Chap. 7, pp. 135–144.

[75] James McLurkin and Jennifer Smith. “Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots”. In: *in 7th International Symposium on Distributed Autonomous Robotic Systems (DARS)*. 2004.

[76] Dirk Helbing, Joachim Keltsch, and Peter Molnar. “Modeling the Evolution of Human Trail Systems”. In: *Nature* 388 (Aug. 1997), pp. 47–50. DOI: 10.1038/40353.

[77] Ferrari G.L. De Nicola R. and Pugliese R. “KLAIM: a kernel language for agents interaction and mobility”. In: *IEEE Transactions on Software Engineering* (1998).

[78] Heiko Hamann, Thomas Schmickl, and Karl Crailsheim. “Self-organized pattern formation in a swarm system as a transient phenomenon of non-linear dynamics”. In: *Mathematical and Computer Modelling of Dynamical Systems* 18.1 (Feb. 2012), pp. 39–50. DOI: 10.1080/13873954.2011.601418.

[79] Shervin Nouyan et al. “Teamwork in Self-organized Robot Colonies”. In: *Trans. Evol. Comp* 13.4 (Aug. 2009), pp. 695–711. ISSN: 1089-778X. DOI: 10.1109/TEVC.2008.2011746. URL: http://dx.doi.org/10.1109/TEVC.2008.2011746.

[80] C. Ronald Kube and Eric Bonabeau. “Cooperative transport by ants and robots”. In: *Robotics and Autonomous Systems* 30 (2000), pp. 85–101.

[81] 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. New York, NY, USA: ACM, 1987, pp. 25–34. ISBN: 0-89791-227-6. DOI: 10.1145/37401.37406. URL: http://doi.acm.org/10.1145/37401.37406.

[82] Vicsek et al. “Novel Type of Phase Transition in a System of Self-Driven Particles”. In: *Physical Review Letters* 75.6 (Aug. 1995), pp. 1226–1229. DOI: 10.1103/physrevlett.75.1226.

[83] L. Adouane. *Autonomous Vehicle Navigation: From Behavioral to Hybrid Multi-Controller Architectures*. CRC Press, 2016. ISBN: 9781498715591. URL: https://books.google.it/books?id=EgzYCwAAQBAJ.

[84] H. Yu et al. *Safe, Autonomous and Intelligent Vehicles*. Unmanned System Technologies. Springer International Publishing, 2018. ISBN: 9783319973012. URL: https://books.google.it/books?id=Sfd5DwAAQBAJ.

[85] G Yeomans. “Autonomous Vehicles”. In: *Handing Over Control: Opportunities and Risks for Insurance. Lloyd’s, London* (2014).

[86] Daniel J. Fagnant and Kara M. Kockelman. “Motorcycle Use in the United States: Crash Experiences, Safety Perspectives, and Countermeasures”. In: *Journal of Transportation Safety & Security* 7.1 (Oct. 2014), pp. 20–39. DOI: 10.1080/19439962.2014.894164.

[87] Daniel J. Fagnant. “Shared autonomous vehicles: Model formulation, sub-problem definitions, implementation details, and anticipated impacts”. In: *2015 American Control Conference (ACC)*. IEEE, July 2015. DOI: 10.1109/acc.2015.7171124.

[88] James Arbib and Tony Seba. “Rethinking Transportation 2020-2030”. In: *RethinkX* (2017).

[89] Jean-François Bonnefon, Azim Shariff, and Iyad Rahwan. “The social dilemma of autonomous vehicles”. In: *Science* (2016).

[90] James Ng and Thomas Bräunl. “Performance Comparison of Bug Navigation Algorithms”. In: *Journal of Intelligent and Robotic Systems* 50.1 (2007), pp. 73–84. ISSN: 1573-0409. DOI: 10.1007/s10846-007-9157-6. URL: https://doi.org/10.1007/s10846-007-9157-6.

[91] Saurabh Sarkar and Ernest Hall. “Virtual force field based obstacle avoidance and agent based intelligent mobile robot”. In: *Intelligent Robots and Computer Vision XXV: Algorithms, Techniques, and Active Vision, 67640R*. Vol. 6764. 2007. DOI: 10.1117/12.734691.

[92] Bence Kovács et al. “A novel potential field method for path planning of mobile robots by adapting animal motion attributes”. In: *Robotics and Autonomous Systems* 82 (2016), pp. 24–34. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2016.04.007. URL: http://www.sciencedirect.com/science/article/pii/S0921889016302159.

[93] Javier Minguez, Florent Lamiraux, and Jean-Paul Laumond. “Motion Planning and Obstacle Avoidance”. In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 827–852. ISBN: 978-3-540-30301-5. DOI: 10.1007/978-3-540-30301-5_36. URL: https://doi.org/10.1007/978-3-540-30301-5_36.

[94] Khadija El Hamidi et al. “Quadcopter attitude and altitude tracking by using improved PD controllers”. In: *International Journal of Nonlinear Dynamics and Control* 1.3 (2019), p. 287. DOI: 10.1504/ijndc.2019.098688.

[95] M Islam, M Okasha, and M M Idres. “Dynamics and control of quadcopter using linear model predictive control approach”. In: *IOP Conference Series: Materials Science and Engineering* 270 (Dec. 2017), p. 012007. DOI: 10.1088/1757-899x/270/1/012007.

[96] Chingiz Hajiyev and Sìtkì Yenal Vural. “LQR Controller with Kalman Estimator Applied to UAV Longitudinal Dynamics”. In: *Positioning* 04.01 (2013), pp. 36–41. DOI: 10.4236/pos.2013.41005.

[97] Ted G. Lewis. “Cognitive stigmergy: A study of emergence in small-group social networks”. In: *Cognitive System Research* 21 (2013), pp. 7–21. ISSN: 1389-0417. DOI: 10.1016/j.cogsys.2012.06.002.

[98] Francesco Ginelli. “The Physics of the Vicsek model”. In: *European Physical Journal: Special Topics* 225 (2016), pp. 2099–2117. ISSN: 1951-6355. DOI: 10.1140/epjst/e2016-60066-8.

[99] Raul Martinez et al. “Collective behavior of Vicsek particles without and with obstacles⋆”. In: *The European Physical Journal E* 41.8 (Aug. 2018). DOI: 10.1140/epje/i2018-11706-8.

[100] Siddharth Mayya et al. “Localization in Densely Packed Swarms Using Interrobot Collisions as a Sensing Modality”. In: *IEEE Transactions on Robotics* 35.1 (Feb. 2019), pp. 21–34. DOI: 10.1109/tro.2018.2872285.

[101] Haitao Zhao et al. “Self-Adaptive Collective Motion of Swarm Robots”. In: *IEEE Transactions on Automation Science and Engineering* 15.4 (Oct. 2018), pp. 1533–1545. DOI: 10.1109/tase.2018.2840828.

[102] G. Indiveri. *Kinematic time-invariant control of a 2D nonholonomic vehicle.* DOI: 10.1109/cdc.1999.831231.

[103] Joshua Cherian Varughese et al. “Quantification and Analysis of the Resilience of Two Swarm Intelligent Algorithms”. In: *3rd Global Conference on Artificial Intelligence*. EasyChair, 2017. DOI: 10.29007/5fhn.

[104] Barbara Arbanas et al. “Aerial-ground robotic system for autonomous delivery tasks”. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2016. DOI: 10.1109/icra.2016.7487759.

[105] Robin R Murphy. “A decade of rescue robots”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012.

[106] AJ Murphy, MJ Landamore, and RW Birmingham. “The role of autonomous underwater vehicles for marine search and rescue operations”. In: *Underwater Technology* 27.4 (2008), pp. 195–205.

[107] Anìbal Matos et al. “Multiple robot operations for maritime search and rescue in euRathlon 2015 competition”. In: *OCEANS 2016-Shanghai*. 2016.

[108] Marco Bibuli et al. “Swarm-based path-following for cooperative unmanned surface vehicles”. In: *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* 228.2 (Jan. 2014), pp. 192–207. DOI: 10.1177/1475090213516108.

[109] Marco Bibuli et al. “A Two Layered Optimal Approach towards Cooperative Motion Planning of Unmanned Surface Vehicles in a Constrained Maritime Environment”. In: *IFAC-PapersOnLine* 51.29 (2018), pp. 378–383. DOI: 10.1016/j.ifacol.2018.09.458.

[110] Joel Esposito, Matthew Feemster, and Erik Smith. “Cooperative manipulation on the water using a swarm of autonomous tugboats”. In: *2008 IEEE International Conference on Robotics and Automation*. IEEE, May 2008. DOI: 10.1109/robot.2008.4543414.

[111] D. Braganza, M. Feemster, and D. Dawson. “Positioning of Large Surface Vessels using Multiple Tugboats”. In: *2007 American Control Conference*. IEEE, July 2007. DOI: 10.1109/acc.2007.4282954.

[112] Van Phuoc Bui et al. “Nonlinear observer and sliding mode control design for dynamic positioning of a surface vessel”. In: *12th International Conference on Control, Automation and Systems*. 2012.

[113] Yoav Naveh, Pinhas Z Bar-Yoseph, and Yoram Halevi. “Nonlinear modeling and control of a unicycle”. In: *Dynamics and Control* 9.4 (1999), pp. 279–296.