UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Dottorato di Ricerca in Automatica e Ricerca Operativa – XIX Ciclo

Tesi di dottorato / Ph.D. Thesis

Optimization Strategies for Constrained Control Systems and Robotic Networks

Ph.D. Advisor: Prof. Ruggero Frezza

Ph.D. Co-Advisor: Prof. John Hauser

Ph.D. Co-Advisor: Prof. Francesco Bullo

Coordinator: Prof. Augusto Ferrante

Ph.D. Candidate: Giuseppe Notarstefano

Padova, March 31, 2007

a Mariella

Alle persone speciali – le più avventurose – non rimangono che due possibilità: o impegnarsi in qualcosa che faccia loro correre dei rischi a livello fisico – come entrare nelle forze aeree speciali, lanciarsi nello spazio o attraversare a piedi il Polo Sud – o esplorare nuove idee, creare forme d'arte, inventare nuove tecnologie, di conseguenza cambiare lo stile di vita di noi tutti.[1]

Desmond Morris

[1]Unfortunately, I was not able to find the English version of this sentence and I did not want to guess a translation

Abstract

In the first part of the work we develop a strategy to compute feasible trajectories of state-input constrained nonlinear control systems. This strategy is interesting itself in understanding the behavior of the system, especially in critical conditions, and represents a useful tool that can be used, in a receding horizon scheme, to perform trajectory tracking in presence of constraints. The strategy is based on a novel optimization technique, introduced by Hauser, to find a regularized solution for *pointwise* constrained optimization of trajectory functionals. We demonstrate the effectiveness of the proposed strategy. In particular, we prove that for suitable choice of the constraints, the solution of the regularized optimization problem exists. The novel concept of *operating region* is another contribution of the work. Its importance relies on the property of providing "a priori" a region where a control engineer can choose (feasible) trajectories that are ensured to be exponentially stabilizable. We prove some preliminary results in the direction of characterizing the operating region of nonlinear control-affine systems driven by (essentially) bounded inputs. The strategy for computation of feasible trajectories is applied to the PVTOL (Planar Vertical Take Off and Landing) aircraft, a simplified model of a real aircraft that captures the main features and challenges of the real model.

In the second part of the work we study two different classes of constrained optimization problems for robotic networks. First, we address the connectivity maintenance problem in wireless networks of robotic agents with double integrator dynamics. We establish an existence theorem for this problem by defining a novel state-dependent graph. Also, we design a distributed "flow-control" algorithm to compute optimal connectivity-maintaining controls. Second, we identify a novel class of optimization problems, namely a networked version of abstract linear programming. For such problems we propose distributed algorithms for networks with various connectivity and/or memory constraints. Finally, we show how various minimum time formation control problems can be tackled through appropriate geometric examples of abstract linear programs.

Keywords: Nonlinear optimal control, constrained optimization, trajectory tracking, Receding Horizon, PVTOL, multi-agent coordination, robotic networks, connectivity, abstract linear programming, formation control.

Sommario

Nella prima parte del lavoro viene sviluppata una strategia per il calcolo di traiettorie ammissibili di un sistema di controllo non lineare con vincoli puntuali sullo stato e sul controllo. La strategia è utile sia per capire il comportamento del sistema, specialmente in condizioni limite, sia in uno schema di controllo predittivo per effettuare inseguimento di traiettorie in presenza dei suddetti vincoli. La strategia è basata su una nuova tecnica di ottimizzazione sviluppata da Hauser. Tale tecnica permette di trovare una soluzione regolarizzata di un problema di controllo ottimo con vincoli. Viene dimostrata l'efficacia della strategia. In particolare, si dimostra che, per una opportuna scelta dei vincoli, la soluzione del problema di ottimo esiste. Il nuovo concetto di *regione operativa* rappresenta un ulteriore contributo. La sua importanza sta nel fatto che permette ad un ingegnere di scegliere una traiettoria, sapendo che questa sarà stabilizzabile esponenzialmente. Vengono dimostrati alcuni risultati preliminari da utilizzare, in futuro, per caratterizzare la regione operativa per sistemi non lineari affini nel controllo guidati da ingressi limitati. La strategia descritta in precedenza è applicata al PVTOL (Planar Vertical Take Off and Landing), un modello semplificato di un aereo reale che ne cattura le caratteristiche principali.

Nella seconda parte della tesi vengono studiate due classi di problemi di ottimizzazione per reti di robot. Per primo, viene affrontato il problema di mantenere la connettività in una rete di robot con dinamica del secondo ordine (doppio integratore) e comunicazione wireless. Viene dimostrato un teorema di esistenza basato sull'introduzione di un nuovo grafo (dipendente dallo stato). Successivamente viene identificata una nuova classe di problemi di ottimizzazione. In pratica si tratta di una versione su reti di problemi di "programmazione lineare astratta". Vengono proposti algoritmi distribuiti per reti con vari vincoli di connettività e di memoria. Infine, si mostra come vari problemi di controllo in formazione in tempo minimo possono essere formulati come problemi di programmazione lineare astratta.

Parole chiave: Controllo ottimo non lineare, ottimizzazione vincolata, inseguimento di traiettoria, controllo predittivo, PVTOL, coordinazione di sistemi multi-agente, reti di robot, connettività, programmazione lineare, controllo in formazione.

Contents

Introduction

Motivations of the work

In the last years many efforts have been done in robotics and aerospace to design autonomous systems that may substitute humans in performing dangerous or very accurate tasks, e.g., exploration, environmental monitoring or surgery. A new emerging concept in designing autonomous systems is to substitute one complex system with a network of simpler subsystems equipped with local communication capabilities. This should provide robustness to the system and improve its performance. The emerging discipline of motion coordination starts from this new concept to study how, many control systems may interact to obtain global performances.

It is clear that two possible points of view arise in the study of autonomous systems. On one hand, we could pay our attention on the single system and look for control strategies that allow to execute more complex tasks. On the other hand, we could try to study the global network in order to obtain an improvement by cooperation.

In this dissertation we follow both these two perspectives, trying to answer, from different points of view and by use of different tools, to the following question. May we find strategies that allow the system (a single agent or a cooperative network) to execute a task, while satisfying some design constraints? And may we do it according to some optimal criteria?

As regards the study of the single control system, we focus on the exploration and tracking of trajectories of the system, while satisfying state and input constraints. Such constraints, we will call them *operating conditions*, may arise from diverse causes such as physical bounds on the states and the inputs or presence of regions where the model is valid or where "bad" phenomena (like loss of controllability) are ensured not to appear. This means that not only we look for trajectories (curves satisfying the dynamics), but furthermore we need *feasible trajectories*, that is, trajectories lying in a region where the operating conditions are satisfied. Our first objective is to find strategies to compute feasible trajectories. In order to develop such strategies we use techniques for the optimization of trajectory functionals developed by Hauser in the last ten years. The main references are [28], [27], [25] and [29]. The second objective is to characterize a region of the state-input space, such that any trajectory remaining in it is uniformly linearly controllable. That is, we ask the linearization of the nonlinear control system about any of these trajectories to be uniformly controllable. We call such region an *operating region*. The importance of characterizing it relies on the fact that, with this region in hand, a control engineer knows a priori that any trajectory in it is, in fact, exponentially stabilizable. We believe that the exploration strategy (to find feasible trajectories) combined with the characterization of an operating region may provide a tool of great engineering interest.

As regards robotic networks, we focus our attention on two main tasks: maintaining network connectivity and reaching a desired formation.

The first task is motivated by the following consideration. In order to ensure a desired emergent behavior (e.g., formation), it is necessary that the group does not disintegrate into subgroups that are unable to communicate to each other. In order to do that, restrictions must be applied on the controls of the agents. In [4], a connectivity constraint was developed for a group of agents modeled as first-order discrete time dynamic systems. We study this problem for a group of agents modeled as second order discrete time dynamic systems, with bounded controls and wireless communication.

The formation task for mobile agents is strongly inspired by nature, where, e.g., swarms or flocks are examples of complex emerging behaviors just based on local information. We analyze the problem of reaching a formation on simple geometric shapes in minimum time. We focus on rendezvous (the simplest formation), and on formation to a line and a circle. The study of this problem, even for the centralized solution, results in studying a constrained optimization problem. The constraints are represented by the positions of the agents. Therefore, starting from the formation task, we study a more general problem. That is, find a class of constrained optimization problems that can be solved by a network, in a distributed way. We found the class of abstract linear programs to be an interesting one. This is a generalized version of linear programming that was introduced by Matoušek, Sharir and Welzl in [49] and extended by Gärtner in [21]. Abstract linear programming is applicable also to some geometric optimization problems, such as the minimum enclosing ball, the minimum enclosing stripe and the minimum enclosing annulus.

These geometric optimization problems are relevant in the design of efficient robotic algorithms for minimum-time formation control problems.

Main contributions

As regards the study of nonlinear control systems, the contributions are threefold. First, we develop a strategy to explore feasible trajectories of constrained nonlinear systems. This strategy is based on techniques for the optimization of trajectory functionals developed by Hauser. The most recent technique allows to regularize constrained optimal control problems by adding a parameterized barrier functional. Second, we provide results showing that, for suitable choice of the state-input constraints, the solution of the regularized optimization problem exists and satisfies second order sufficiency conditions. The proof relies on an elegant application of the implicit function theorem. Third, we provide a preliminary characterization of an operating region, that is a subset of the state-input space where every trajectory (remaining in it) is guaranteed to be exponentially stabilizable.

As regards the motion coordination part, the contributions are threefold. For the connectivity maintenance problem, we establish an existence theorem by introducing a novel state-dependent graph. We also design distributed "flow-control" algorithm to compute optimal connectivity-maintaining controls. Second, we identify a novel class of distributed optimization problems, namely a networked version of abstract linear programming. For such problems we propose distributed algorithms for networks with various connectivity and/or memory constraints. Third, we apply the previous distributed computation problems in minimum-time formation control problems, such as the rendezvous problem and the problems of line and circle formations.

Thesis outline

Chapter 1 is a review of the trajectory tracking projection operator. Through the use of this tool, a dynamically constrained optimization problem may be converted in an unconstrained problem. The projection operator based Newton method is described. This technique is the basis for the strategy developed in Chapter 2.

In Chapter 2 we provide a new strategy to compute feasible trajectories of constrained control systems. We start providing a formal definition of the task that we want to perform. We describe the exploration strategy and prove the correctness of the strategy for suitable values of the constraints. Finally, we briefly discuss how, under suitable stabilizability assumptions, a receding horizon technique may be implemented to track feasible trajectories.

In Chapter 3 we apply the exploration strategy to the simplified PVTOL aircraft. After introducing the PVTOL model we show how to compute trajectories of the unconstrained PVTOL given a desired trajectory of the center of gravity. We describe two strategies to compute feasible trajectories of the PVTOL and show some simulations.

In Chapter 4 we introduce the novel concept of *operating region*. It is a region where any trajectory is uniformly linearly controllable. We provide a review of the main controllability notions. Then, we characterize an operating region for feedback linearizable systems, and for control-affine systems driven by sampled controls. We also prove that state trajectories generated by sampled controls converge uniformly to actual state trajectories.

In Chapter 5 we introduce the mathematical model that we will use in the next chapters to deal with networks. After a brief review on graph definitions and main properties, we introduce a model for a network of processors. Then, we define a more complex network model, where the nodes are dynamical systems, e.g. mobile robots.

In Chapter 6 we address the connectivity maintenance problem in wireless networks of robotic agents with double integrator dynamics. We establish an existence theorem for this problem by defining a novel state-dependent graph. Also, we design a distributed “flow-control” algorithm to compute optimal connectivity-maintaining controls.

In Chapter 7 we introduce a network version of abstract linear programming. We start introducing abstract linear programs after a short review on linear programming. Then we define network abstract linear programs and state the three distributed algorithms. We prove correctness of the algorithms and establish halting conditions.

In Chapter 8 we introduce the problem of minimum time formation for a robotic network. We focus on the formation control problem for a point formation (rendezvous), a line formation and a circle formation. We show that they can be formulated as network abstract linear programs. A control and communication law based on the distributed algorithm of the previous chapter is proposed as an approximate solution.

Chapter 1

Projection operator

This chapter is a review of [27] and [25]. A trajectory tracking nonlinear projection operator is introduced as a powerful tool for the optimization of trajectory functionals. Through the use of the projection operator, a dynamically constrained optimization problem is converted into an unconstrained problem that is solved by use of a Newton method. This novel and effective tool is at the basis of the strategy for computation of feasible trajectories of constrained systems presented in Chapter 2.

1.1 Introduction

We are interested in studying the nature of the set of trajectories of a nonlinear control system described by an ordinary differential equation on $\mathbb{R}^n$ with inputs lying in $\mathbb{R}^m$. A *trajectory* of f is a curve $\eta = (x, u)$ in $L_\infty[0, \infty)$ such that

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

for all $t \geq 0$ where f is a C^r mapping of $\mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ with $1 \leq r \leq \infty$. With this definition, we explicitly require *trajectories* to be *bounded* curves that satisfy the differential equation. Although we will be mostly interested in trajectories on the finite horizon $[0, T]$, it is often useful to consider a finite length trajectory as a portion of one of infinite extent.

A trajectory $\xi = (\alpha, \mu)$ is called *exponentially stabilizable* if (and only if) there is a feedback law $u(t) = k(t, x(t), \alpha(t), \mu(t))$ (with $k(t, \alpha(t), \alpha(t), \mu(t)) = \mu(t)$ for all t) such that α is an exponentially stable (state) trajectory of the closed loop system

$$\dot{x}(t) = f(x(t), k(t, x(t), \alpha(t), \mu(t))) \ .$$

That is, there exist $M < \infty$, $\lambda > 0$, and $\delta > 0$ such that $\|x(t) - \alpha(t)\| \leq Me^{-\lambda(t-t_0)}\|x(t_0) - \alpha(t_0)\|$ for all $t \geq t_0 \geq 0$ and all $x(t_0)$ such that $\|x(t_0) - \alpha(t_0)\| < \delta$. We would also impose some smoothness and boundedness conditions on k. Since a C^1 nonlinear system is exponentially stable if and only if its linearization is, we may restrict our attention to feedbacks of the form

$$u(t) = \mu(t) + K(t)(\alpha(t) - x(t))$$

so that ξ is exponentially stabilizable if and only if there is a bounded gain matrix K that stabilizes the linearization of f about ξ. That is, the linear system (without input)

$$\dot{z}(t) = A_c(t)z(t) := [A(\xi(t)) - B(\xi(t))K(t)]z(t)$$

is exponentially stable where $A(\xi(t)) := D_1 f(\alpha(t), \mu(t))$ and $B(\xi(t)) := D_2 f(\alpha(t), \mu(t))$. Thus, the state transition matrix Φ associated with A_c satisfies $\|\Phi(t, \tau)\| \leq Me^{-\lambda(t-\tau)}$. We will say that "$K$ stabilizes ξ" to indicate this property.

Let $\mathcal{T}$ denote the set of exponentially stabilizable trajectories of f. Clearly $\mathcal{T}$ is a subset of L_∞. A fundamental property proved in [27], that we recall in this chapter, is that $\mathcal{T}$ is, in fact, a C^r Banach Manifold. Moreover, the tangent vectors at a given trajectory are precisely the exponentially stabilizable trajectories of the linearization of the system about the given trajectory. The key tool in the analysis is a projection operator $\mathcal{P}$ that results when the above type of feedback is used to stabilize a trajectory. Using a family of projection operators, an atlas of coordinate charts for $\mathcal{T}$ may be constructed. These results provide important insights into the trajectory exploration problem. In particular, they allow us to use vector space operations [42] to effectively explore the trajectory manifold. Moreover, the analysis presented here for trajectories on the infinite time interval $[0, \infty)$ lays the foundation for practical calculations on finite intervals.

Notation

In this dissertation we will often deal with linear vector spaces. In particular we will consider metric vector spaces (e.g., Banach spaces) and also spaces with an inner product (e.g., Hilbert spaces). Given a function $f : X \to Y$, where X and Y are two normed spaces, f *bounded* means that there exists $\bar{f} \in \mathbb{R}$ such that $\|f(x)\|_Y \leq \bar{f}$ for every x in the domain of f (dom f). Clearly we let $\|\cdot\|_Y$ denote the norm in the topology of Y. If f is Fréchet differentiable, we denote with $Df(\bar{x})$ the derivative of f evaluated at $\bar{x} \in X$. If $x = (x_1, x_2) \in X$, we will use either $D_{x_1}f(\bar{x}_1, \bar{x}_2)$ or $D_1 f(\bar{x}_1, \bar{x}_2)$ to emphasize the differential of f with respect to the variable x_1, evaluated at $\bar{x} = (\bar{x}_1, \bar{x}_2) \in X$. As above, we will use ξ to refer to a curve in $L_\infty[0, \infty)$ and $\xi(t)$ to refer to a point on the curve. To measure the size of things $\|z\| = \sup_{[0,\infty)} \|z(t)\|$ where $\|z(t)\|$ is the Euclidean norm on $\mathbb{R}^{n+m}$. Also, we denote $\phi(t; x_0, u(\cdot))$ the solution of the control system at instant t, starting from the state x_0 and driven by the control $u(\cdot)$. For a linear system, we denote $\Phi(t, \tau)$, $\{t, \tau\} \subset \mathbb{R}_+ \cup \{0\}$, the state transition matrix. When we want to underline that the state transition matrix is associated to the linearization of a nonlinear system about the trajectory ξ, we will use the notation $\psi_t(\xi)$, where $\psi_\tau(\xi) \cdot (t) = \Phi_\xi(t, \tau)$. Finally, $\|\Phi(t, \tau)\|$ is the matrix norm (on $\mathbb{R}^{n \times n}$) induced by the chosen vector norm.

1.2 Trajectory tracking Projection Operator

In this section we introduce a *projection operator* $\mathcal{P}$ that results from the use of a trajectory tracking control law and analyze its differentiability properties.

Suppose that K stabilizes $\xi_0 \in \mathcal{T}$ and consider the mapping $\mathcal{P} : \xi = (\alpha, \mu) \mapsto \eta = (x, u)$ defined by

$$\begin{aligned} &\dot{x}(t) = f(x(t), u(t)), && x(0) = \alpha(0), \\ &u(t) = \mu(t) + K(t)(\alpha(t) - x(t)) . \end{aligned} \tag{1.1}$$

Clearly $\xi_0 = \mathcal{P}(\xi_0)$ since ξ_0 is a stable (by K) trajectory of f. Moreover, since ξ_0 is an exponentially stable trajectory (of the closed loop system f, K), $\mathcal{P}$ is well defined on an L_∞ neighborhood of ξ_0, i.e., $\eta = \mathcal{P}(\xi)$ is an exponentially stable trajectory for *any curve* ξ such that $\|\xi - \xi_0\|$ is sufficiently small. That is, $\mathcal{P}$ maps curves ξ into trajectories η. We shall see that $\mathcal{P}$ is locally onto $\mathcal{T}$. We call $\mathcal{P}$ a projection operator since $\mathcal{P} = \mathcal{P}^2 := \mathcal{P} \circ \mathcal{P}$ which follows from the fact that every trajectory is a fixed point of $\mathcal{P}$. Note that the specific behavior of $\mathcal{P}$ depends on the choice of K. When it is desirable to emphasize this dependence, we will write $\mathcal{P}_K$.

We are now ready to explore the differentiability properties of $\mathcal{P}$. The following lemma is a key tool for the proof of the main result. The proof can be found in [27].

Lemma 1.1 *Consider the nonlinearly perturbed linear system*

$$\dot{y}(t) = A(t)y(t) + B(t)\zeta(t) + R(t, y(t), \zeta(t)), \qquad y(0) = \beta(0)$$

where $\zeta = (\beta, \nu)$, A is exponentially stable ($\|\Phi(t,\tau)\| \leq Me^{-\lambda(t-\tau)}$), B is bounded ($\|B(t)\| \leq b$), and R is higher order (in its last two arguments, uniformly in t). The mapping $\zeta \mapsto y$ is continuous at $\zeta = 0$ with

$$\|y\| \leq [M(1 + b/\lambda) + \epsilon(\delta)]\ \|\zeta\|$$

for all $\|\zeta\| < \delta$ where ϵ is a continuous nondecreasing function with $\epsilon(0) = 0$.
□

The following result is fundamental. It shows that $\mathcal{P}$ (defined on the infinite horizon) is Fréchet differentiable with respect to the L_∞ norm. We report the proof of the theorem because it clarifies the structure of the derivative of $\mathcal{P}$, that is the linear operator $D\mathcal{P}(\xi)$, which plays a key role in the treatment.

Theorem 1.2 (Differentiability of $\mathcal{P}$, [27]) *Suppose that f is C^r, $1 \leq r \leq \infty$, $\xi_0 \in \mathcal{T}$, and K stabilizes ξ_0. Then $\mathcal{P} = \mathcal{P}_K$ is C^r in a neighborhood of ξ_0.*

Proof: As noted above, by exponential stability, $\mathcal{P}$ is well defined for all ξ sufficiently close to ξ_0. Let ξ be such a curve and set $\eta = \mathcal{P}(\xi)$.

Step 1. $\mathcal{P}$ is continuous at ξ.
Let $\xi = (\alpha, \mu)$ and $\zeta = (\beta, \nu)$ and set $\mathcal{P}(\xi) = (x, u)$ and $\mathcal{P}(\xi + \zeta) = (x + y, u + w)$. Then $y(0) = \beta(0)$ and

$$\begin{aligned}\dot{y}(t) &= f(x(t) + y(t), u(t) + w(t)) - f(x(t), u(t)) \\ &= A_c(t)y(t) + B_c(t)\zeta(t) + R_f(t, y(t), w(t))\end{aligned}$$

where $A_c(t) = A(\eta(t)) - B(\eta(t))K(t)$, $B_c(t) = B(\eta(t))[K(t)\ I]$, and $R_f(t, y(t), w(t))$ is the (obvious) remainder. Since R_f is higher order in $(y(t), w(t))$ and $w(t) = -K(t)y(t) + [K(t)\ I] \cdot (\beta(t), \nu(t))$, we see that R_f is higher order in $(y(t), \zeta(t))$ which we will write as $R(t, y(t), \zeta(t))$. The y dynamics are thus governed by

$$\dot{y}(t) = A_c(t)y(t) + B_c(t)\zeta(t) + R(t, y(t), \zeta(t)),\ y(0) = \beta(0)\ .$$

Continuity of $\mathcal{P}$ at ξ follows from Lemma 1.1.

Step 2. $\mathcal{P}$ is differentiable at ξ with $D\mathcal{P}(\xi) \cdot \zeta = (z, v)$ given by

$$\begin{aligned} \dot{z}(t) &= A(\eta(t))z(t) + B(\eta(t))v(t),\ z(0) = \beta(0), \\ v(t) &= \nu(t) + K(t)[\beta(t) - z(t)]\ . \end{aligned} \tag{1.2}$$

Differentiability requires that $y - z$ depends on ζ in a higher order fashion. Since z satisfies

$$\dot{z}(t) = A_c(t)z(t) + B_c(t)\zeta(t),\ z(0) = \beta(0),$$

we see that

$$(\dot{y} - \dot{z})(t) = A_c(t)(y - z)(t) + R(t, \zeta),\ (y - z)(0) = 0,$$

where R satisfies $\|R(t, \zeta)\| \le r(\|\zeta\|)\|\zeta\|$ with r continuous, nondecreasing, and $r(0) = 0$. Here we have used that fact that y can be linearly bounded with respect to ζ (see Step 1). Using Lemma 1.1 we see that

$$\frac{\|(y - z)(t)\|}{\|\zeta\|} \le \frac{M}{\lambda} r(\|\zeta\|)$$

so that $\mathcal{P}$ is differentiable.

Step 3. Higher derivatives.
The derivative of $\mathcal{P}$ defines a linear parameter varying (LPV) system $\mathcal{L}(\eta) = D\mathcal{P}(\xi)$. Here, the bounded linear operator $\mathcal{L}(\eta)$ depends on the *parameter* η. Differentiability properties of LPVs are proven in [27]. Using these results along with the chain rule, it may shown by induction that $\mathcal{P}$ is C^r. ■

Remark 1.3 (Importance of using L_∞ norm) *It is important to note that $\mathcal{P}$ may not be differentiable if the another norm is used, e.g., the L_2 norm.*□

As expected the derivative of the projection operator $\mathcal{P}$ is the linear projection operator $D\mathcal{P}(\xi)$ given by the standard linearization. Note that the character of $D\mathcal{P}(\xi)$ depends *only* on the *trajectory* $\eta = \mathcal{P}(\xi)$ and not on the particular $\xi = \mathcal{P}^{-1}(\eta)$. This implies the fundamental property $D\mathcal{P}(\mathcal{P}(\xi)) = D\mathcal{P}(\xi)$.

We have already seen that $\mathcal{P}$ is a projection with $\mathcal{P}^2 = \mathcal{P}$. Using the chain rule and the property just stated above, it is easy to see that, given a bounded curve (not necessarily a trajectory!) ξ,

$$D\mathcal{P}(\xi) = D[\mathcal{P}(\mathcal{P}(\xi))] = D\mathcal{P}(\mathcal{P}(\xi)) \cdot D\mathcal{P}(\xi) = D\mathcal{P}(\xi) \cdot D\mathcal{P}(\xi)$$

so that $D\mathcal{P}(\xi)$ is also a projection. That $D\mathcal{P}(\xi)$ is a projection can be seen directly from the differential equations (1.2) that define it. Indeed, $\gamma = D\mathcal{P}(\xi) \cdot \zeta$ is a trajectory of the linearization at ξ. The form of the feedback law in (1.2) (exactly the same as for the nonlinear system) implies that *any* trajectory of the linearization at ξ is a fixed point of $D\mathcal{P}(\xi)$, i.e.,

$$\gamma = D\mathcal{P}(\xi) \cdot \gamma = D\mathcal{P}(\xi) \cdot D\mathcal{P}(\xi) \cdot \zeta \, .$$

Since this holds for every ζ, we see that $D\mathcal{P}(\xi) = (D\mathcal{P}(\xi))^2$.

An explicit computation of $\mathcal{P}$ derivatives up to third order may be found in [25].

Given $\xi_0 \in \mathcal{T}$, we can use tangent vectors $\zeta \in T_{\xi_0}\mathcal{T}$ to parametrize $\xi \in \mathcal{T}$ near ξ_0 according to $\mathcal{P}(\xi_0 + \zeta)$. Clearly, such points lie in $\mathcal{T}$ for small ζ. In fact, every $\xi \in \mathcal{T}$ near ξ_0 can be expressed this way, uniquely.

Proposition 1.4 (Local charts near trajectories, [27]) *Suppose that $\xi_0 \in \mathcal{T}$ and let $\mathcal{P}$ be a projection operator defined in a neighborhood of ξ_0 (e.g., $\mathcal{P} = \mathcal{P}_K$ with ξ_0 stabilized by K). Then*

$$\mathcal{Q}_{\xi_0}(\zeta) := \mathcal{P}(\xi_0 + \zeta)$$

is a C^r diffeomorphism of a neighborhood of the origin in $T_{\xi_0}\mathcal{T}$ onto a neighborhood of ξ_0 in $\mathcal{T}$. □

Using this property a $\mathcal{C}^r$ atlas of charts, indexed by trajectories $\xi \in \mathcal{T}$, is available. It is easy to see that these charts are $\mathcal{C}^r$ compatible. This proves the following important theorem.

Theorem 1.5 ([27]) *$\mathcal{T}$ is a $\mathcal{C}^r$ Banach manifold.* □

1.3 Projection Operator Newton method

In this section we present the Newton method for optimization of trajectory functionals introduced in [25].

We are interested in optimal control problems (OCPs) of the form

$$\begin{aligned} &\text{minimize } \int_0^T l(\tau, x(\tau), u(\tau))\, d\tau + m(x(T)) \\ &\text{subject to } \dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \end{aligned} \tag{1.3}$$

over the class of (essentially) bounded inputs. This problem is often referred to as an *unconstrained* optimal control problem since, under uniqueness conditions and for fixed x_0, the state trajectory is completely determined (on its interval of existence) by the choice of control $x(t) \equiv x(t; u(\cdot))$ allowing one to remove the dynamic constraint, writing the objective as a function of $u(\cdot)$ alone. (Such a *shooting* approach is, of course, not recommended.)

We are mainly interested in objectives and systems that possess a certain degree of smoothness: let $l(t, x, u)$, $m(x)$, and $f(x, u)$ be (at least) C^3 in x and u (with $l(t, x, u)$, e.g., continuous in t). To ensure that solutions (should they exist) of the optimal control are nice (and somewhat likely), we desire some convexity conditions. We require the set $f(x, \mathbb{R}^m) \subset \mathbb{R}^n$ to be convex for each $x \in \mathbb{R}^n$. We also require the pre-Hamiltonian to be strongly convex in u, that is, the map

$$u \mapsto l(t, x, u) + p^T f(x, u) =: H^-(t, x, u, p)$$

is strictly convex for all $(t, x, p) \in \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^n$, possessing a second derivative matrix that is uniformly positive definite. This ensures a unique control $\bar{u}^*(t, x, p)$ that minimizes the pre-Hamiltonian providing a C^2 (in (x, p)) Hamiltonian $H(t, x, p) := H^-(t, x, \bar{u}^*(t, x, p), p)$. This property is satisfied when, e.g., $f(x, u)$ is affine in u and l is quadratic (and positive definite for $t \in [0, T]$) in u. To the purpose of existence, we expect the terminal cost m to be nonnegative (and preferably proper). With sufficient conditions of f, l, and m, one may guarantee existence of optimal trajectories, see, e.g., [39, 10]. Also of interest here are techniques from the direct methods of the calculus of variations—see [9] for an accessible introduction.

Let X denote the closed subspace of $L_\infty^{n+m}[0, T]$ of curves $\zeta = (\beta, \nu)$ with continuous β, $\beta(0) = 0$, and bounded ν. Equipped with the norm $\|\zeta\|_X = \|\zeta\|_{L_\infty}$, X is a Banach space. Define $\pi_1 := [I\ 0]$ and $\pi_2 := [0\ I]$ so that $\beta = \pi_1 \zeta$ and $\nu = \pi_2 \zeta$. Trajectories of f through x_0 belong to the affine subspace $\widetilde{X} := (x_0, 0) + X$. Defining the functional

$$\begin{aligned} h(\xi) &:= \int_0^T l(\tau, \alpha(\tau), \mu(\tau))\, d\tau + m(\alpha(T)) \\ &= \int_0^T l(\tau, \xi(\tau))\, d\tau + m(\pi_1 \xi(T)) \end{aligned}$$

for curves $\xi = (\alpha, \mu) \in \widetilde{X}$, we see that the optimal control problem (1.3) is equivalent to the constrained optimization problems

$$\min_{\xi \in \mathcal{T}} h(\xi) = \min_{\xi = \mathcal{P}(\xi)} h(\xi)$$

where the constraint set $\mathcal{T}$ is a Banach submanifold of $\widetilde{X}$. Defining

$$g(\xi) := h(\mathcal{P}(\xi))$$

for $\xi \in \mathcal{U} \subset \widetilde{X}$ with $\mathcal{P}(\mathcal{U}) \subset \mathcal{U} \subset \operatorname{dom} \mathcal{P}$, we see that the optimization problems

$$\min_{\xi \in \mathcal{T}} h(\xi) \quad \text{and} \quad \min_{\xi \in \mathcal{U}} g(\xi)$$

are equivalent in the following sense. If $\xi^* \in \mathcal{T} \cap \mathcal{U}$ is a constrained local minimum of h, then it is an unconstrained local minimum of g. If $\xi^+ \in \mathcal{U}$ is an unconstrained local minimum of g in $\mathcal{U}$, then $\xi^* = \mathcal{P}(\xi^+)$ is a constrained local minimum of $\mathcal{T}$. This observation is the basis for the development of a family of quasi-Newton descent methods for the optimization of h over $\mathcal{T}$.

As seen in the previous section the projection operator $\mathcal{P}$ provides a convenient parametrization of the trajectories in the neighborhood of a given trajectory. Indeed, the tangent space $T_\xi \mathcal{T}$ of bounded trajectories of the linearization of $\dot{x} = f(x, u)$ about $\xi \in \mathcal{T}$ can be used to parametrize *all* nearby trajectories [27]. That is, given $\xi \in \mathcal{T}$, there is an $\epsilon > 0$ such that, for each $\eta \in \mathcal{T}$ with $\|\eta - \xi\| < \epsilon$ there is a *unique* $\zeta \in T_\xi \mathcal{T}$ such that $\eta = \mathcal{P}(\xi + \zeta)$. (Of course, there are many other curves $\tilde{\xi} \in \mathcal{U}$ such that $\eta = \mathcal{P}(\tilde{\xi})$.) Note also that $\zeta \mapsto D\mathcal{P}(\xi) \cdot \zeta$ is the bounded *linear* projection operator defined by linearizing (1.1) about ξ and that $\zeta \in T_\xi \mathcal{T}$ if and only if $\zeta = D\mathcal{P}(\xi) \cdot \zeta$.

Notice that in Section 1.2, following [27], these results were proven for exponentially stabilizable trajectories defined on the infinite time-interval $[0, +\infty)$. For finite-interval trajectories, stabilizability is not relevant, but the differentiability properties of $\mathcal{P}$ still holds as the properties stated above. The proof will be given in a forthcoming paper by Hauser.

The Newton method for the optimization of trajectory functionals is the following.

```
Algorithm (projection operator Newton method)
Given initial trajectory $\xi_0 \in \mathcal{T}$
For $i = 0, 1, 2...$
    design $K$ defining $\mathcal{P}$ about $\xi_i$
    search direction
        $\zeta_i = \arg \min_{\zeta \in T_\xi \mathcal{T}} Dg(\xi_i) \cdot \zeta + \frac{1}{2} D^2 g(\xi_i)(\zeta, \zeta)$    (1.4)
    step size
        $\gamma_i = \arg \min_{\gamma \in (0,1]} g(\xi + \gamma \zeta_i);$    (1.5)
    project
        $\xi_{i+1} = \mathcal{P}(\xi_i + \gamma_i \zeta_i).$    (1.6)
end
```

This algorithm is quite similar to the usual Newton method for unconstrained optimization of a function $g(\cdot)$ (e.g., in finite dimensions). As usual, the second order Taylor polynomial is used as a quadratic model function for determining a descent direction. A pure Newton method would, of course, use a fixed step size of $\gamma_i = 1$—the line search is common for expanding the region of convergence. The key differences are that (i) the search direction minimization (1.4) is performed on the tangent space to the trajectory manifold and (ii) the update (1.6) *projects* each iterate on to the trajectory manifold. The algorithm is easily generalized (or globalized) by replacing the Newton direction calculation (1.4) by a quasi-Newton search direction calculation

$$\zeta_i = \arg \min_{\zeta \in T_{\xi_i} \mathcal{T}} Dg(\xi_i) \cdot \zeta + \tfrac{1}{2}\, q(\xi_i) \cdot (\zeta, \zeta) \tag{1.7}$$

where $q(\xi_i)$ is a suitable positive definite (to be defined below) approximation to $D^2 g(\xi_i)$.

Note also that it is not necessary to do an accurate line search as expressed in 1.5. The standard *backtracking* line search (see, e.g., [7]) works quite well.

Chapter 2

Exploration and tracking of feasible trajectories

In this chapter we provide a new strategy to explore *feasible trajectories* of nonlinear systems, that is to find curves that satisfy the dynamics as well as pointwise state-input constraints. This strategy is interesting itself in understanding the behavior of the system especially in critical conditions and represents a useful tool that can be used to perform trajectory tracking in presence of constraints. The strategy is based on a novel optimization technique, introduced by Hauser, to find a regularized solution for *pointwise* constrained optimization of trajectory functionals.

2.1 Introduction

Output tracking is a challenging task in the control of nonlinear systems. It has interesting practical applications in several fields as aerospace, robotics and automotive. Given an output reference trajectory, the control objective is to find a feedback law such that the output asymptotically tracks the reference trajectory. Some of the most known techniques to solve this and related control problems may be found for example in [35], [38] and [18]. A more recent technique for solving trajectory tracking (and also output tracking) is Receding Horizon control. An excellent survey of various approaches to this technique is [50]. A non exhaustive literature review includes [14], [37], [36], [24].

A preliminary step or a different point of view in the solution of the tracking problem is the exploration of the trajectory manifold of the system, that is the characterization of all output trajectories that can be tracked and the parametrization of the state-input trajectories with respect to the output ones. An important contribution to the solution of this problem has been done by Devasia et al. in [16] and successive works.

Having in mind engineering applications, there is an important aspect to take into account in the solution of exploration and tracking, that is the presence of constraints in the system. Such constraints, we will call them *operating conditions*, may arise from diverse causes such as physical bounds on the states and the inputs or presence of regions where the model is valid or where "bad" phenomena (like loss of controllability) are ensured not to appear. This means that not only we look for trajectories (curves satisfying the dynamics), but furthermore we need *feasible trajectories*, that is, trajectories lying in a region where the operating conditions are satisfied.

A possible way to take into account constraints (especially the input ones) is to modify the trajectory tracking approach into path following. This point of view has been largely used in robotics and was formalized in [26]. In this approach the system dynamics are re-parameterized so that the tangential velocity along the curve is used as a control input. This provides an extra degree of freedom in the designer's hands to satisfy the constraints. However, this approach strongly relies on the geometric structure of the systems and provides only one extra degree of freedom. The use of this approach starting with a naive choice of the desired trajectory could result in a significant loss of performance, since the velocity profile on the curve is no more assigned. Moreover, it is not even guaranteed to be successful. There could be situations in which the velocity (the extra input) has itself strong constraints. For example, in controlling a rigid aircraft the velocity cannot get down a reasonable positive value. In other words the path following approach does not protect very much if the choice of the desired trajectory is naive.

We attack the problem of exploring feasible trajectories by using optimal control. The strategy consists of the following steps. Given a desired output trajectory, first, a state-input trajectory consistent with the output is computed as in [53]. Then, a feasible trajectory, close in the L_2 norm to the desired one, is computed by solving a regularized version of an optimal control problem with pointwise constraints. This regularization is based on the introduction of a parameterized barrier functional, to a standard L_2 functional, that takes into account the constraints. The parameter allows us to decide the "level" of feasibility of the trajectory, that is how close this is to the boundary imposed by the operating conditions. The regularization idea for solving constrained optimal control was introduced in [29]. This effective technique is just based on the regularization of a trajectory functional optimization with pointwise constraints and then on its solution by means of the projection operator based Newton method described in Chapter 1. We demonstrate the effectiveness of the proposed strategy. In particular we provide results showing that for suitable choice of the constraints and the design parameters, the solution of the regularized optimal control problem exists and satisfies second order sufficiency conditions. In the next chapter we show that for the PVTOL example the solution behaves very well even in presence of relatively tight constraints.

In Section 2.2 we provide a formal definition of the task that we want to perform. We also review two techniques for trajectory exploration of unconstrained systems. In Section 2.3 we describe the barrier functional based exploration strategy. Section 2.4 provides the existence results mentioned above. In Section 2.5 we briefly show how, under suitable stabilizability assumptions, a receding horizon technique may be implemented to track the feasible trajectory. Section 2.6 contains a discussion of future perspectives.

2.2 Exploration task definition and trajectory morphing

In this section we provide a formal definition of what we shall refer to as *exploration task* for a nonlinear control system. Then, we review two techniques to compute a desired (state-input) trajectory of an unconstrained nonlinear system, provided a desired task is given.

Before defining the exploration task, we suggest the following consideration. The feedback can significantly improve the performance of a system, allowing it, for example, to work in unstable regions. However, it cannot enlarge the range of system trajectories. This means that, in general, it is not suitable to choose arbitrary curves and ask the system to track them.

Usually in the definition of a tracking task, the "final" objective is that a portion of the state follows a desired curve. We call this portion of the state *objective output* to distinguish from the measured output. In this dissertation we always consider state feedback, thus assuming that the whole state may be measured. Therefore in the future we will often omit the prefix objective before output.

If we consider a vehicle, the desired objective could be that the center of gravity of the vehicle follows a desired curve with an assigned velocity profile.

Clearly, the dynamics of the vehicle will put severe constraints on the class of curves that can be tracked.

Remark 2.1 *Here, we are not considering (pointwise) constraints of states and inputs, due, for example, to the existence of safety regions or to actuators saturation. We are only taking into account (dynamic) constraints imposed by the dynamics.* □

In order to guarantee that the desired output curve is not completely unrelated to the trajectories of the system, we generate the desired curve as the output of a nonlinear system that captures some of the features of the target system. This choice relies on the following consideration. Very often engineers have intuitive notions of dynamics of complicated systems and it is common to have on hand simplified models that capture many important features of a system.

In the following we define formally the task and task achievement. Consider the *target control system* Σ

$$\begin{aligned} \dot{x}(t) =& f(x(t), u(t)), \\ y(t) =& p(x(t)), \quad t \geq 0 \end{aligned} \tag{2.1}$$

where $f(x,u)$ and $p(x)$ are $\mathcal{C}^r$ mappings, $r \geq 2$, $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ the control input and $y \in \mathbb{R}^p$ is the objective output, used to define the task.

The task is defined by the tuple $(y_d(\cdot), \Sigma_0)$, where y_d is the *desired output curve* and Σ_0 is the *task control system* defined as

$$\begin{aligned} \dot{x}(t) =& f_0(x(t), u(t)), \\ y(t) =& p(x(t)), \quad t \geq 0 \end{aligned} \tag{2.2}$$

where $f_0(x,u)$ is $\mathcal{C}^r$, $r \geq 2$, $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$ and $y \in \mathbb{R}^p$.

Remark 2.2 (Properties of the task control system) *The task control system is often chosen to be a "special" system, that is, a system for which the control engineer has an intuitive understanding of interesting class of trajectories. For example we may look for differential flatness or "nice" geometry.* □

Remark 2.3 (Examples of task control systems) *An example of task system could be the target system without the actuator dynamics. That is, the task system has the same dynamics as the target system except for the* *actuator transfer function that is taken to be the identity. In flight control, we could consider a "complicated" model of an aircraft as target system, and the simplified coordinated flight model as task system.* □

We assume the desired output curve $y_d(\cdot)$ to be an *output trajectory* of the task system Σ_0, that is, there exists $\xi_0 = (x_0(\cdot), u_0(\cdot))$ such that $\dot{x}_0(t) = f_0(x_0(t), u_0(t))$ (ξ_0 is a trajectory of Σ_0) and $y_d(\cdot) = p(x_0(\cdot))$.

Given a desired output curve (candidate output trajectory) $y_d(\cdot)$, we say that $\xi_d = (x_d(\cdot), u_d(\cdot))$ is a lifted trajectory of $y_d(\cdot)$ if $\dot{x}_d(t) = f(x_d(t), u_d(t))$ (ξ_d is a trajectory of Σ) and $y_d(\cdot) = p(x_d(\cdot))$. If such ξ_d exists, we call $y_d(\cdot)$ an output trajectory.

If we find such ξ_d, we say that the task has been *exactly achieved.* If we can find a trajectory $\xi_p = (x_p(\cdot), u_p(\cdot))$ of Σ, such that for any $\epsilon > 0$, $\|h(x_p(\cdot)) - y_d(\cdot)\| < \epsilon$ (for some norm, e.g., the L_2 or the L_∞ norms), we say that the task has been *practically achieved.*

If the task system, assigned with the task, is exactly the target system, the exploration strategy described in next section may be directly applied. However, in many practical applications, this is not the case. In the following we review two techniques, introduced in [28] and [30] respectively, to find lifted trajectories of unconstrained nonlinear systems.

One way of obtaining a trajectory that practically achieves the task is to choose a weighted L_2 norm for the distance. That is we may ask to solve the optimal control problem with quadratic cost

$$\begin{aligned}
\text{minimize} \quad & \frac{1}{2}\int_0^T \|x(\tau) - x_0(\tau)\|_Q^2 + \|u(\tau) - u_0(\tau)\|_R^2 \, d\tau + \frac{1}{2}\|x(T) - x_0(T)\|_{P_1}^2 \\
\text{subj. to} \quad & \dot{x}(t) = f(x(t), u(t)), \quad t \in [0, T] \\
& x(0) = x_0
\end{aligned} \tag{2.3}$$

or in the equivalent form

$$\min_{\xi \in \mathcal{T}} h(\xi) = \min_{\xi = \mathcal{P}(\xi)} h(\xi)$$

where $\xi = (x(\cdot), u(\cdot))$ and $h(\xi)$ is the quadratic cost functional, $\mathcal{T}$ is the trajectory manifold of the (target) system and $\mathcal{P}$ is the projection operator introduced in Chapter 1.

The problem in (2.3) is an "instantaneous morph" in that we attempt to change $\xi_0 = (x_0(\cdot), u_0(\cdot))$ into a nearby trajectory of Σ in one step. This forces to deal with the vector field $f(x + x_0(t)) - f_0(x_0(t), u_0(t))$, which is not desirable because of the large difference between f and f_0.

The first method, introduced in [28], is a *homotopy method.* We consider the λ parameterized family of optimization problems

$$\begin{aligned}
\text{minimize} \quad & \frac{1}{2}\int_0^T \|x(\tau) - x_0(\tau)\|_Q^2 + \|u(\tau) - u_0(\tau)\|_R^2 \, d\tau + \frac{1}{2}\|x(T) - x_0(T)\|_{P_1}^2 \\
\text{subj. to} \quad & \dot{x}(t) = f_\lambda(x(t), u(t)), \quad t \in [0, T] \\
& x(0) = x_0
\end{aligned}$$

where $f_\lambda(x,u) = \lambda f(x,u) + (1-\lambda) f_0(x,u)$, $\lambda \in [0,1]$. That is, we use a vector field homotopy f_λ combined with the optimization, to obtain a homotopy curve of local minimizers $\xi_\lambda = (x_\lambda(\cdot), u_\lambda(\cdot))$. For $\lambda = 0$, we know the solution $\xi_0 = (x_0(\cdot), u_0(\cdot))$. Also, for $\lambda > 0$, small, we know that ξ_λ varies in a smooth manner since ξ_0 is a second order sufficient minimizer (for $Q > 0$, $R > 0$, $P_1 >= 0$).

Remark 2.4 (Design of the weighting matrices) *Note that, since our primary objective is the output to be close to the desired curve $y_d(\cdot)$, we design the matrices Q, R and P_1 to weight the outputs more than the other states. This aspect will be discussed better in the next section.* □

The second method, introduced in [30], is a *dynamics embedding* technique, that relies on embedding the (target) system into a "fully actuated" system. More formally, given $(x_0(\cdot), u_0(\cdot))$, we augment the target system so that we can solve, for each $t \geq 0$,

$$\dot{x}_0(t) = f(x_0(t), u_0(t)) + G u_{\text{ext}}(t)$$

for $u_{\text{ext}}(t)$. We choose G to make this possible, providing an appropriate notion of "fully actuated". Then, $(x_0(t), u_0(t), u_{\text{ext}}(t))$, $t \geq 0$, provides an initial trajectory of the augmented nonlinear system, from which we may begin the following optimization

$$\begin{aligned}
\text{minimize} \quad & \frac{1}{2}\int_0^T \|x(\tau) - x_0(\tau)\|_Q^2 + \|u(\tau) - u_0(\tau)\|_R^2 + r\|u_{\text{ext}}(\tau)\|^2 \, d\tau \\
& \quad + \frac{1}{2}\|x(T) - x_0(T)\|_{P_1}^2 \\
\text{subj. to} \quad & \dot{x}(t) = f(x(t), u(t)) + G u_{\text{ext}}(t), \quad t \in [0, T] \\
& x(0) = x_0.
\end{aligned}$$

Here, $r > 0$ is a weighting parameter that is initially small and then increased as far as it is sufficiently larger than the norm of Q, R and P_1. This provides a good approximate trajectory of the target system.

Remark 2.5 (Scenarios for dynamics embedding) *The dynamics embedding technique is well suited for those systems where the input directly affects the dynamics of the objective output. This is the case, for example, of many vehicles, such as the simplified motorcycle model for which the technique was introduced in [30]. In the next chapter we apply this technique to the PVTOL for which the same consideration holds.* □

2.3 Computing feasible trajectories by use of trajectory optimization

As seen in the previous section, we know strategies for generating trajectories of the system given a desired output. However, it can easily happen that the lifted trajectory does not satisfy the operating conditions. In other words while we have useful tools to parameterize the unconstrained trajectories of the system with respect to the desired output trajectories, we do not have a similar tool to impose the operating conditions being satisfied. Therefore we can informally state our objective.

> *Objective [informal description]* Develop a tool that, given a desired output trajectory, provides a feasible trajectory (state-input), that meets the operating conditions and is such that its output part is "close" to the desired one.

Clearly, regarding the objective, we need to define formally the notion of "closeness" (between the desired and the feasible outputs) and the notion of feasible trajectory. As in the previous section, we use a weighted L_2 norm between the desired trajectory (lifted from the desired output) and the feasible one. Given a desired trajectory ξ_d (lifted from a desired output), we set the distance of a trajectory ξ from ξ_d to

$$h(\xi) = \int_0^T \frac{1}{2}\|x(\tau) - x_d(\tau)\|_Q^2 + \frac{1}{2}\|u(\tau) - u_d(\tau)\|_R^2 \, d\tau + \frac{1}{2}\|x(T) - x_d(T)\|_{P_1}^2$$

where Q, R and P_1 are positive definite matrices. The reason we decide to fully penalize the state and the input will be clear in the following when we state the optimal control problem.

The operating conditions are taken into account defining the following region in the state-input space

$$\overline{\mathcal{XU}_\rho} = \{(x,u) \in \mathbb{R}^n \times \mathbb{R}^m | c_j(x,u;\rho) \le 0, \rho > 0, \ j = 0,1,...,k\}$$

where each $c_j(x,u;\rho)$ is $\mathcal{C}^2$ or better in x and u and varies smoothly with the parameter ρ. Furthermore, the following assumption holds.

Operating Conditions Assumption (OCA). The region of operating conditions satisfies the following properties:

(i) for any $\rho > 0$, $\mathcal{XU}_\rho$, the interior of $\overline{\mathcal{XU}_\rho}$, is a nonempty connected set.

(ii) for any $0 < \rho_1 < \rho_2$, $\mathcal{XU}_{\rho_1} \subset \mathcal{XU}_{\rho_2}$;

(iii) the projection of $\mathcal{XU}_\rho$ on the input space

$$\pi_u \mathcal{XU}_\rho = \{u \in \mathbb{R}^m | (x,u) \in \mathcal{XU}_\rho \ \forall \text{ fixed } x\}$$

is convex;

(iv) for every desired output trajectory $y_d(\cdot)$, the lifted trajectory $\xi_d = (x_d(\cdot), u_d(\cdot))$ (if it exists) is such that $\exists\ \rho_0 > 0$ such that $(x_d(t), u_d(t)) \in \mathcal{XU}_{\rho_0}$ for every $t \in [0,T]$. $\square$

Roughly speaking the operating conditions describe a region where the space and the input must lie at every instant. For theoretical purpose we parameterize this region with a scaling factor $\rho > 0$ that allows us to expand and shrink the region in such a way that we can always entirely include the desired trajectory in it.

Now it is clear what we mean by feasible trajectory. A trajectory $\xi = (x(\cdot), u(\cdot))$ is said a *feasible trajectory* for Σ with respect to the region $\overline{\mathcal{XU}_\rho}$, if it is a trajectory of Σ and is entirely contained in $\overline{\mathcal{XU}_\rho}$. If ξ is entirely contained in $\mathcal{XU}_\rho$ it is said a *strictly feasible trajectory.*

We are ready to introduce the exploration strategy. We start defining the barrier functional associated with the region $\mathcal{XU}_\rho$, that is

$$b_{\delta_c,\rho}(\xi) = \int_0^T \sum_j \beta_{\delta_c}(-c_j(\tau, \alpha(\tau), \mu(\tau); \rho)) \tag{2.4}$$

where $z \mapsto \beta_{\delta_c}(z)$ is defined as

$$\beta_{\delta_c}(z) = \begin{cases} -\log z & z > \delta_c \\ \frac{k-1}{k}\left[\left(\frac{z-k\delta_c}{(k-1)\delta_c}\right)^k - 1\right] - \log \delta_c & z \le \delta_c. \end{cases}$$

where $k > 1$ is an even integer. In practice, we have found $k = 2$ to work well.

This barrier functional is the key element of the new optimization technique introduced in [29]. In fact, the $\mathcal{C}^2$ function $\beta_{\delta_c}(\cdot)$ retains many of the important properties of the usual log barrier function $z \mapsto \log(z)$ (used in finite dimension constrained optimization), while expanding the domain of finite values from $]0, \infty[$ to $]-\infty, +\infty[$. This ensures that the barrier functional may be evaluated on any trajectory and not only on feasible ones. Moreover, both functions defining $\beta_{\delta_c}(\cdot)$ are (strictly) convex and strictly decreasing on their domains. Thus, if $c_j : \mathbb{R} \to \mathbb{R}$ is a strictly convex proper function, the function $z \mapsto \beta_{\delta_c}(-c_j(z))$ is also a strictly convex proper function on $\mathbb{R}$.

With this definition in hand we can define the modified cost functional

$$h_{\epsilon_c,\rho}(\xi) = h(\xi) + \epsilon_c b_{\delta_c,\rho}(\xi)$$

and consistently

$$g_{\epsilon_c,\rho}(\xi) = h_{\epsilon_c,\rho}(\mathcal{P}(\xi)).$$

The exploration strategy is described in the table.

Given

$(y_d(\cdot), \Sigma_0)$, an output trajectory and a task system

$\mathcal{XU}_\rho$, $\rho > 0$, a set of operating conditions

find $\xi_d \in \mathcal{T}$, such that $y_d(\cdot) = h(x_d(\cdot))$ (or its *practical* version)

solve

$$\begin{aligned} \text{minimize} \quad & h_{\epsilon_c,\rho}(\xi) = h(\xi) + \epsilon_c b_{\delta_c,\rho}(\xi) \\ \text{subj. to} \quad & \xi = \mathcal{P}(\xi) \end{aligned}$$

or the equivalent unconstrained version

$$\text{minimize} \quad g(\xi) = h_{\epsilon_c}(\mathcal{P}(\xi))$$

tune

Q, R, P_1	*shaping* parameters
$\epsilon_c, \delta_c, \rho$	*closeness* parameters

We call Q, R and P_1 shape parameters because they are used to weight some components of the (state-control) trajectory more than others. In our strategy we penalize the output much more than the other states and the inputs. The reason we impose $Q > 0$, $R > 0$ and $P_1 > 0$ is that this allows to have second order sufficiency (SSC) of the minimum. The closeness parameters are used to set the operating conditions (ρ) and to tune the level of closeness of the feasible trajectory from the boundary of the operating conditions (ϵ_c, δ_c).

Remark 2.6 (Engineering goal in searching feasible trajectories) *It is important to underline that our goal is not to find the closest trajectory satisfying the operating conditions, that is, we do not want to solve a constrained optimal control problem. The approach we follow is an engineering point of view, in the sense that we want a tool that provides feasible trajectories and where the "closeness" can be tuned by a "knob".* □

2.4 Continuity of the minimizer with respect to parameters

In the following we present results on the existence of solutions of the optimal control problem for suitable values of the operating conditions.

Next two theorems state that, starting with a feasible trajectory ξ_d for some ρ_0, we can perturb the cost $h(\xi)$ with a small ϵ_c so that the optimization problem is still solvable. Then, fixing ϵ_c, it is possible to decrease ρ of a certain amount (thus enforcing the operating conditions) and still have a solution.

Theorem 2.7 (Continuity of the minimizer w.r.t. ϵ_c) *Let $\rho_0 > 0$ be such that $\xi_d = (x_d(\cdot), u_d(\cdot))$ is a strictly feasible trajectory, that is, $\xi_d \in \mathcal{XU}_{\rho_0}$. Then there exist $\epsilon > 0$, such that for each $|\epsilon_c| < \epsilon$*

$$\min_{\xi \in \mathcal{T}} h_{\epsilon_c, \rho_0}(\xi)$$

has an SSC local minimizer ξ_{ϵ_c} near ξ_d.Furthermore $\epsilon_c \mapsto \xi_{\epsilon_c}$ is continuously differentiable. □

Theorem 2.8 (Continuity of the minimizer w.r.t. ρ) *Let $\rho_0 > 0$, $\epsilon_c > 0$ be such that $\xi_0 \in \mathcal{T}$ is an SSC local minimizer of $g_{\epsilon_c, \rho_0}(\xi)$. Then, there is an $r > 0$ such that for each ρ, $|\rho - \rho_0| < r$, there is a local SSC minimizer ξ_ρ of $g_{\epsilon_c, \rho}(\xi)$ near ξ_0. Furthermore $\rho \mapsto \xi_\rho$ is continuously differentiable.* □

The proof of the two theorems is based on the same arguments stated in the following.

We consider the situation where the cost functional depends smoothly on a finite dimensional parameter and the system is independent of the parameter. In this section, we will refer to this parameter as $\rho \in \mathbb{R}^p$. This parameter may include, for instance, the scalar parameter ρ used for specifying the size of the feasible region as well as the scalar parameter ϵ used in determining strictly feasible trajectories. We will suppose that the scaling and offsets of the parameters have been chosen in such a manner that the nominal value of the parameter vector is $\rho = 0$.

We thus write

$$g_\rho(\xi) = h(\mathcal{P}(\xi), \rho).$$

Let $\xi_0 \in \mathcal{T}$ and consider the nature of $g_0(\xi)$ on a neighborhood of ξ_0. In particular, we consider ξ of the form $\xi_0 + \zeta$ where $\|\zeta\| < \delta$ and $\delta > 0$ is such that $\xi_0 + \zeta \in \operatorname{dom} \mathcal{P}$ for each such ζ. For $\mathcal{C}^2$ $g_0(\cdot)$, we have

$$g_0(\xi_0 + \zeta) = g_0(\xi_0) + Dg_0(\xi_0) \cdot \zeta + \frac{1}{2} D^2 g_0(\xi_0) \cdot (\zeta, \zeta) + r(\xi_0, \zeta) \cdot (\zeta, \zeta) \quad (2.5)$$

where the remainder satisfies

$$|r(\xi_0, \zeta) \cdot (\zeta, \zeta)| / \|\zeta\|^2 \to 0 \quad \text{as} \quad \|\zeta\| \to 0 \quad (2.6)$$

where $\| \cdot \|$ is the L_∞ norm. Using the $\mathcal{C}^2$ identity

$$\phi(1) = \phi(0) + \phi'(0) + \int_0^1 (1 - s)\, \phi''(s)\, ds$$

together with $\phi(s) = g_0(\xi_0 + s\zeta)$, we obtain the explicit expression

$$r(\xi_0, \zeta) \cdot (\zeta_1, \zeta_2) = \int_0^1 (1 - s) \left[D^2 g_0(\xi_0 + s\zeta) - D^2 g_0(\xi_0) \right] ds \cdot (\zeta_1, \zeta_2) \quad (2.7)$$

which has been slightly generalized to depend on three, possibly independent, perturbations. Using the fact that $D^2 g_0(\cdot)$ is continuous as a mapping from the trajectory manifold $\mathcal{T}$ to set of continuous bilinear functionals on L_∞, we easily verify that the remainder $r(\xi_0, \zeta) \cdot (\zeta, \zeta)$ defined by (2.7) satisfies, as it *must*, the higher order property (2.6). Equations (2.5), (2.7) provide a *second order expansion with remainder* formula for the $\mathcal{C}^2$ mapping $g_0(\cdot)$, valid in an L_∞ neighborhood of any $\xi_0 \in \mathcal{T}$. In fact, the formula given by (2.5), (2.7) is somewhat more general, requiring only that $\xi_0 \in L_\infty$ and $\delta > 0$ are such that $B_\delta(\xi_0) \subset \operatorname{dom} \mathcal{P}$.

Now, since the functional $g_0(\cdot)$ is the composition of an integral functional and a projection operator, the value of the bilinear expression $D^2g_0(\xi)\cdot(\zeta_1,\zeta_2)$ for $\xi \in \operatorname{dom} \mathcal{P}$ and $\zeta_i \in L_\infty$ is of the form

$$D^2g_0(\xi)\cdot(\zeta_1,\zeta_2) = \int_0^T \gamma_1(\tau)^T W(\tau)\gamma_2(\tau)\, d\tau + (\pi_1\gamma_1(T))^T P_1(\pi_1\gamma_2(T))$$

where $\gamma_i = D\mathcal{P}(\xi)\cdot\zeta_i$ and where $P_1 = P_1^T \in \mathbb{R}^{n\times n}$ and the bounded matrix $W(t) = W(t)^T \in \mathbb{R}^{n\times n}$, $t \in [0,T]$, depend continuously on $\eta = \mathcal{P}(\xi)$, hence continuously on ξ. Using these facts, we see that

Lemma 2.9 *Let $\xi_0 \in \mathcal{T}$ and suppose that $\delta > 0$ is such that $B_\delta(\xi_0) \subset$ dom $\mathcal{P}$. Then, there is a nondecreasing function $\bar{r}(\cdot)$ with $\bar{r}(0) = 0$ such that*

$$|r(\xi_0,\zeta)\cdot(\zeta_1,\zeta_2)| \le \bar{r}(\|\zeta\|)\, \|\zeta_1\|_{L_2}\|\zeta_2\|_{L_2} \tag{2.8}$$

for all $\zeta, \zeta_1, \zeta_2 \in B_\delta$.

Proof: Since $\mathcal{P}$ is $\mathcal{C}^2$, $D\mathcal{P}(\xi)$ is a continuous linear projection operator with respect to the L_∞ norm. Using an explicit formula for $D\mathcal{P}(\xi)\cdot\zeta$, it is easy to see that $D\mathcal{P}(\xi)$ may be extended to a linear projection operator $D\mathcal{P}(\xi)_{L_2}$ on L_2 that is continuous with respect to the L_2 norm. The result follows easily using (2.7). ■

Suppose now that $\xi_0 \in \mathcal{T}$ is a stationary trajectory of $g_0(\cdot)$ so that $Dg_0(\xi_0)\cdot\zeta = 0$ for all $\zeta \in L_\infty$ and that $\delta > 0$ is such that $B_\delta(\xi_0) \subset \operatorname{dom} \mathcal{P}$. It follows that

$$g_0(\xi_0 + \zeta) \ge g_0(\xi_0) + \frac{1}{2}D^2g_0(\xi_0)\cdot(\zeta,\zeta) - \bar{r}(\|\zeta\|)\, \|\zeta\|_{L_2}^2 \tag{2.9}$$

for all $\|\zeta\| < \delta$ where $\bar{r}(\cdot)$ is given by Lemma 2.9. Restricting (2.9) to $\zeta \in T_{\xi_0}\mathcal{T}$, we obtain the fundamental second order sufficient condition (SSC) for ξ_0 to be an isolated local minimizer.

Theorem 2.10 *Suppose $\xi_0 \in \mathcal{T}$ is such that $Dg_0(\xi_0)\cdot\zeta = 0$ for all $\zeta \in L_\infty$ and that there is a $c_0 > 0$ such that*

$$D^2g_0(\xi_0)\cdot(\zeta,\zeta) \ge c_0\|\zeta\|_{L_2}^2 \quad \text{for all} \quad \zeta \in T_{\xi_0}\mathcal{T}\,. \tag{2.10}$$

Then ξ_0 is an isolated local minimizer in the sense that there is a $\delta > 0$ such that

$$g_0(\xi_0) < g_0(\xi)$$

for all $\xi \in \mathcal{T}$ with $\|\xi - \xi_0\| < \delta$, $\xi \ne \xi_0$.

Proof: Taking $\delta_1 > 0$ be such that $\bar{r}(\delta_1) < c_0/4$, we find that $g_0(\xi_0 + \zeta) \geq g_0(\xi_0) + (c_0/4)\|\zeta\|_{L_2}^2$ for all $\zeta \in T_{\xi_0}\mathcal{T}$ with $\|\zeta\| < \delta_1$. Recall that each $\xi \in \mathcal{T}$ near ξ_0 can be represented by a unique $\zeta \in T_{\xi_0}\mathcal{T}$ according to $\xi = \mathcal{P}(\xi_0 + \zeta)$ and that the mapping $\xi \mapsto \zeta$ is continuous. Thus there is a $\delta < \delta_1$ such that $\|\xi - \xi_0\| < \delta$ implies that $\|\zeta\| < \delta_1$. The result follows. ■

We call a local minimizer $\xi_0 \in \mathcal{T}$ satisfying (2.10) a *second order sufficient condition local minimizer*, SSC local minimizer for short. According to theorem 2.10, *every* SSC local minimzer is an *isolated* local minimizer. We also note that, in words, the condition (2.10) says that the quadratic functional $\zeta \mapsto D^2 g_0(\xi_0) \cdot (\zeta, \zeta)$ is *strongly positive* on the subspace $T_{\xi_0}\mathcal{T}$.

Consider now the (local) minimization of $g_\rho(\xi)$ as the parameter ρ is varied on a neighborhood of $\rho = 0$ where ξ_0 is known to be an SSC local minimizer of $g_0(\xi)$. Since $D^2 g_\rho(\xi)$ is continuous in both ξ and ρ, we expect that, for each sufficiently small ρ, there will be a corresponding SSC local minimizer ξ_ρ near ξ_0 and that the mapping $\rho \mapsto \xi_\rho$ will be continuous, and perhaps differentiable. The key idea is to use an appropriate implicit function theorem (IFT) to *solve* the first order necessary condition equation

$$Dg_\rho(\xi_\rho) = 0 \tag{2.11}$$

for ξ_ρ as a function of ρ starting from ξ_0 at $\rho = 0$. Proceeding formally, we differentiate (2.11) with respect to ρ to obtain

$$\frac{\partial}{\partial \rho} Dg_\rho(\xi_\rho) + D\{Dg_\rho(\xi_\rho)\} \cdot \xi_\rho' = 0\,.$$

Thus, the derivative of ξ_ρ with respect to ρ, ξ_ρ', if it exists, is given formally by

$$\xi_\rho' = -[D\{Dg_\rho(\xi_\rho)\}]^{-1} \cdot \frac{\partial}{\partial \rho} Dg_\rho(\xi_\rho)\,.$$

In this case, we might expect that there is an implicit function theorem that says something like, if $D\{Dg_\rho(\xi_\rho)\}$ is invertible at $\rho = 0$, then there is a neighborhood of $\rho = 0$ on which $\rho \mapsto \xi_\rho$ is well defined and C^1. In what sense should the operator $D\{Dg_0(\xi_0)\}$ be invertible and how can it be ensured? It turns out that the appropriate condition is that $D^2 g_0(\xi_0)$ be strongly positive on $T_{\xi_0}\mathcal{T}$, i.e., that it satisfy (2.10).

Theorem 2.11 *Suppose that $\xi_0 \in \mathcal{T}$ is an SSC local minimizer of $g_0(\xi)$. Then, there is a $\delta > 0$ such that, for each ρ such that $\|\rho\| < \delta$, there is a local SSC minimizer ξ_ρ of $g_\rho(\xi)$ near ξ_0. Furthermore $\rho \mapsto \xi_\rho$ is continuously differentiable.*

Proof: The key is to show that, for ρ sufficiently small, we can compute a $\xi \in \mathcal{T}$ such that $Dg_\rho(\xi) \cdot \zeta = 0$ for all $\zeta \in L_\infty$. We proceed by parametrizing $\xi \in \mathcal{T}$ locally by $\gamma \in T_{\xi_0}\mathcal{T}$ according to $\xi = \mathcal{P}(\xi_0 + \gamma)$ and searching over γ. As in the proof of many IFTs, we solve for the desired γ using a contraction mapping. For simplicity, we will denote $T_{\xi_0}\mathcal{T}$ by X so that we search for $\gamma \in X$ such that $Dg_\rho(\mathcal{P}(\xi_0 + \gamma)) \cdot \zeta = 0$ for all $\zeta \in L_\infty$.

First, note that, since $D^2 g_0(\xi_0)$ is strongly positive on X, the well-defined quadratic minimization problem

$$\lambda = \arg \min_{\zeta \in X} -\omega \cdot \zeta + \frac{1}{2} D^2 g_0(\xi_0) \cdot (\zeta, \zeta)$$

defines a linear mapping $SS : \omega \mapsto \lambda : \text{dom } SS \subset X^* \to X$ for some continuous linear functionals $\omega \in X^*$. The linear mapping SS provides the solution $\lambda \in X$ to the functional equation

$$D^2 g_0(\xi_0) \cdot (\lambda, \zeta) = \omega \cdot \zeta, \quad \zeta \in X,$$

effectively providing an inverse to the operator $D\{Dg_0(\xi_0)\}$ formally described above. We will see that the functionals $\omega \in X^*$ of interest belong to the domain of SS.

Define $\mathcal{F}_\rho : X \to X^*$ by

$$\mathcal{F}_\rho(\gamma) \cdot \zeta = D^2 g_0(\xi_0) \cdot (\gamma, \zeta) - Dg_\rho(\xi_0 + \gamma) \cdot \zeta$$

for all $\zeta \in X$. Note that $\mathcal{F}_\rho(\gamma) \cdot \zeta$ is of the form

$$\mathcal{F}_\rho(\gamma) \cdot \zeta = \int_0^T a(\tau)^T z(\tau) + b(\tau)^T v(\tau) \, d\tau + r_1^T z(T)$$

for $\zeta = (z(\cdot), v(\cdot)) \in X$ where $a(\cdot)$, $b(\cdot)$, and r_1 depend smoothly on the data ρ and γ. It follows that $\mathcal{F}_\rho(\gamma) \in \text{dom } SS \subset X^*$. A straightforward calculation shows that

$$\mathcal{G}_\rho(\gamma) = SS \cdot \mathcal{F}_\rho(\gamma)$$

defines a continuous operator $\mathcal{G}_\rho : X \to X$ that is also continuous in ρ.

Note that, if $\gamma \in X$ is a fixed point of $\mathcal{G}_\rho(\cdot)$, $\gamma = \mathcal{G}_\rho(\gamma)$, then $Dg_\rho(\xi_0 + \gamma) \cdot \zeta = 0$ for all $\zeta \in X$. This will imply that $Dg_\rho(\xi_0 + \gamma) \cdot \zeta = 0$ for all $\zeta \in L_\infty$ provided that $\mathcal{P}(\xi_0 + \gamma)$ is sufficiently near ξ_0. In that case, we conclude that $\xi_\rho = \mathcal{P}(\xi_0 + \gamma)$. Also, for $\rho = 0$, we see that $\gamma = 0$ is the fixed point, $\mathcal{G}_0(0) = 0$ as expected.

We will show that, for ρ sufficiently small, $\mathcal{G}_\rho(\cdot)$ is a contraction mapping with a unique fixed point. For $\rho = 0$, noting that $Dg_\rho(\xi_0+\gamma)\cdot(\cdot) = D^2g_0(\xi_0)\cdot(\gamma,\cdot) + o(\|\gamma\|)$, we see that

$$\mathcal{G}_0(\gamma) = SS \cdot (D^2g_0(\xi_0)\cdot(\gamma,\cdot) - Dg_\rho(\xi_0+\gamma)\cdot(\cdot)) = o(\|\gamma\|)$$

where we have used the fact that SS is continuous (bounded) on the elements of X^* of the noted form. By continuity in ρ, we see that there exist $\rho_1, \delta > 0$ such that

$$\|\mathcal{G}_\rho(\gamma)\| \le \delta$$

whenever $\|\rho\| \le \rho_1$ and $\|\gamma\| \le \delta$. Now, fixing ρ, $\|\rho\| \le \rho_1$,

$$\begin{aligned}\mathcal{G}_\rho(\gamma_1) - \mathcal{G}_\rho(\gamma_2) &= SS \cdot [Dg_\rho(\xi_0+\gamma_2)\cdot(\cdot) - Dg_\rho(\xi_0+\gamma_1)\cdot(\cdot)] \\ &= SS \cdot \left[\int_0^1 D^2g_\rho(\xi_0+\gamma_1+s(\gamma_2-\gamma_1))\,ds\cdot(\gamma_2-\gamma_1,\cdot)\right]\end{aligned}$$

so that there is a $k < \infty$ such that

$$\|\mathcal{G}_\rho(\gamma_1) - \mathcal{G}_\rho(\gamma_2)\| \le k\delta\|\gamma_1 - \gamma_2\|$$

for $\|\gamma_1\| \le \delta$ and $\|\gamma_2\| \le \delta$. Shrinking δ, if necessary, so that $k\delta \le 1/2$, we see that $\mathcal{G}_\rho$ is a contraction with unique fixed point γ_ρ.

To see that $\rho \mapsto \gamma_\rho$ is continuous, write

$$\begin{aligned}\|\gamma_{\rho_1} - \gamma_{\rho_2}\| &= \|\mathcal{G}_{\rho_1}(\gamma_{\rho_1}) - \mathcal{G}_{\rho_2}(\gamma_{\rho_2})\| \\ &\le \|\mathcal{G}_{\rho_1}(\gamma_{\rho_1}) - \mathcal{G}_{\rho_1}(\gamma_{\rho_2})\| + \|\mathcal{G}_{\rho_1}(\gamma_{\rho_2}) - \mathcal{G}_{\rho_2}(\gamma_{\rho_2})\| \\ &\le (1/2)\|\gamma_{\rho_1} - \gamma_{\rho_2}\| + \|\mathcal{G}_{\rho_1}(\gamma_{\rho_2}) - \mathcal{G}_{\rho_2}(\gamma_{\rho_2})\|\end{aligned}$$

so that

$$\|\gamma_{\rho_1} - \gamma_{\rho_2}\| \le 2\|\mathcal{G}_{\rho_1}(\gamma_{\rho_2}) - \mathcal{G}_{\rho_2}(\gamma_{\rho_2})\|$$

showing that $\rho \mapsto \gamma_\rho$ is continuous since $\rho \mapsto \mathcal{G}_\rho(\gamma)$ is continous (for fixed γ).

Differentiability is proven just following the second part of the proof of Theorem 4.E in [65]. ■

2.5 Receding horizon for feasible trajectory tracking

In this section we briefly discuss how we may use the barrier functional based optimization in a receding horizon scheme to perform trajectory tracking.

We assume that we are given a desired *feasible* trajectory of the nonlinear system on the positive infinite time interval.

Calling $(x_{\text{des}}(\cdot), u_{\text{des}}(\cdot))$ the desired *feasible* trajectory, we assume that $(x_{\text{des}}(t), u_{\text{des}}(t)) \in \overline{\mathcal{XU}}$ for $t \geq 0$, where $\overline{\mathcal{XU}}$ is a simply connected compact subset of the state input space. We assume that the linearization is exponentially stabilizable or, in other words, that there exists a bounded matrix function $K(\cdot)$, such that $A(\cdot) - B(\cdot)K(\cdot)$ is exponentially stable.

Under this assumption and the regularity of $f(x, u)$, it can be proven that the Jacobian matrices

$$\begin{aligned} A(t) =& D_1 f(x_{\text{des}}(t), u_{\text{des}}(t)) \\ B(t) =& D_2 f(x_{\text{des}}(t), u_{\text{des}}(t)) \end{aligned}$$

are bounded, that is there exist $M_A > 0$ and $M_B > 0$ such that

$$\forall t \geq 0 \;\; \|A(t)\| < M_A, \;\; \|B(t)\| < M_B.$$

Furthermore, denoting $\tilde{x}(t) = x(t) - x_{\text{des}}(t)$, $\tilde{u}(t) = u(t) - u_{\text{des}}(t)$ and $\tilde{f}(\tilde{x}, \tilde{u}, t) = f(\tilde{x} + x_{\text{des}}(t), \tilde{u} + u_{\text{des}}(t))$, we can prove also that $\tilde{f}_2(\tilde{x}, \tilde{u}, t) = \tilde{f}(\tilde{x}, \tilde{u}, t) - [A(t)\tilde{x} + B(t)\tilde{u}]$ satisfies

$$\lim_{\|(\tilde{x},\tilde{u})\| \to 0} \; \sup_{t \geq 0} \frac{\tilde{f}_2(\tilde{x}, \tilde{u}, t)}{\|(\tilde{x}, \tilde{u})\|} = 0.$$

That is, in a neighborhood of the origin, the nonlinear *error system* can be approximated by its linearization about the origin.

Consider the linear quadratic optimization problem

$$\begin{aligned} \text{minimize} \;\; & \frac{1}{2} \int_t^{t+T} \|z(\tau)\|_Q^2 + \|v(\tau)\|_R^2 \, d\tau + \frac{1}{2} \|z(t+T)\|_{P_1}^2 \\ \text{subj. to} \;\; & \dot{z}(\tau) = A(\tau) z(\tau) + B(\tau) v(\tau), \quad \tau \in [t, t+T] \\ & z(t) = x(t) - x_{\text{des}}(t) \end{aligned} \tag{2.12}$$

where $Q > 0$, $R > 0$ and $P_1 \geq 0$. Let $t \mapsto K_T(t)$ be the feedback of the receding horizon scheme obtained by (2.12).

We make the following standing assumption.

Assumption 2.12 *Given $(x_{\text{des}}(t), u_{\text{des}}(t)) \in \overline{\mathcal{XU}}$, $t \geq 0$, we assume that there exists $T^* > 0$ such that for any $T \geq T^*$, $A(t) - B(t)K_T(t)$, $t \geq 0$, is exponentially stable.*

Now, we consider the nonlinear optimization problem

$$\begin{aligned}\text{minimize } & \int_t^{t+T} \frac{1}{2}\|x(\tau)-x_{\text{des}}(\tau)\|_Q^2 + \frac{1}{2}\|u(\tau)-u_{\text{des}}(\tau)\|_R^2 \\ & \qquad + \epsilon_{\text{c}} \sum_j \beta_{\delta_{\text{c}}}(-c_j(x(\tau),u(\tau);\rho))d\tau + \|x(T)-x_{\text{des}}(T)\|_{P_1} \\ \text{subj. to } & \dot{x}(\tau) = f(x(\tau),u(\tau)), \quad x(t)=x_t\end{aligned}$$

with the same notation defined in the previous section. Here, $\epsilon_{\text{c}} > 0$ and $\delta_{\text{c}} > 0$ are design parameter that need to be fixed.

The nonlinear *receding horizon* scheme that we propose is the following. Solve the finite horizon optimization every $\delta > 0$ seconds and use the optimal control trajectory $u_T^*(t+\tau; x(t), t)$, $\tau \in [0,\delta]$, to drive the system from $x(t)$ at time t to $x_T^*(t+\delta; x(t), t)$ at time $t+\delta$. We denote this receding horizon scheme as $\mathcal{RH}(T,\delta)$

Using Assumption 2.12 and robustness of exponential stability with respect to perturbation and sampling, we may prove local exponential stability of $\mathcal{RH}(T,\delta)$, $T \geq T^*$.

A rigorous proof of this result is behind the goal of this thesis and will be objective of future work.

2.6 Discussion

We studied the problem of exploring feasible trajectories of nonlinear control systems, that is trajectories satisfying state and input constraints. We developed an effective strategy that, in the next chapter, we successfully apply to the simplified PVTOL aircraft model. An important direction of investigation is in the area of Receding Horizon Control. It includes (i) developing a receding horizon scheme, based on the same optimization technique as the exploration strategy, that allows to track feasible trajectories while satisfying state and input constraint, (ii) designing a hierarchical strategy that implements both the exploration and the tracking tasks, and proving the correctness of such strategies.

Chapter 3

The PVTOL example

3.1 Introduction

One of the field where nonlinear control is more active is aerospace. In the last years many effort have been done to design unmanned aerial vehicles (UAV) to perform dangerous and prohibitive tasks. Our objective is to apply the exploration strategy described in the previous chapter and its related receding horizon scheme to the model of a real aircraft. As a preliminary step, we applied the strategy to a simplified system that well models either the longitudinal dynamics of the aircraft or the lateral dynamics. This model is known as PVTOL aircraft. The PVTOL was introduced by Hauser et al. in [32] in order to capture the lateral non-minimum phase behavior of a real aircraft. This model has been widely studied in the literature for its characteristic of combining important features of nonlinear systems with "tractable" equations. Furthermore, the dynamics of many other mechanical systems can be rewritten in a similar fashion, e.g., the cart-pole system, the pendubot [59], the bicycle model [23], [30] and, as said, the longitudinal dynamics of a real aircraft. Since the PVTOL has been introduced in 1992, many researchers have studied this system providing different solutions for trajectory tracking. A non exhaustive literature review of works on the PVTOL includes [32], [46], [3], [44].

We apply the strategy introduced in the previous chapter to the input constrained PVTOL aircraft. First, we show that we can use the decoupled model of the PVTOL as task control system. Starting from trajectories of the decoupled model we show, using results proven in [31], that there exists a trajectory of the target system that has exactly the same output as the decoupled model. We compute an approximation of this trajectory by use of the dynamics embedding technique. Then, we use the barrier functional exploration strategy to compute feasible trajectories of the PVTOL. In particular we use it in two different schemes based on the homotopy and dynamics embedding methods described in the previous chapters.

In Section 3.2 we introduce the PVTOL model. In Section 3.3 we show how to compute trajectories of the unconstrained PVTOL given a desired trajectory of the center of gravity. Section 3.4 describes two strategies to compute feasible trajectories of the PVTOL and in Section 3.5 simulations are shown. In Section 3.6 we discuss future perspectives.

3.2 PVTOL modeling

In this section we present the PVTOL dynamics, its flatness property and its non-minimum phase nature with respect to outputs (center of gravity position) when $\epsilon_{\text{PVTOL}} \neq 0$.

In [32] the model of the PVTOL aircraft was introduced. Using standard aeronautic conventions the equations of motion are given by

$$\begin{aligned} \ddot{y} &= u_1 \sin\varphi - \epsilon_{\text{PVTOL}}\, u_2 \cos\varphi \\ \ddot{z} &= -u_1 \cos\varphi - \epsilon_{\text{PVTOL}}\, u_2 \sin\varphi + g \\ \ddot{\varphi} &= u_2. \end{aligned} \tag{3.1}$$

The aircraft state is given by the position (y, z) of the center of gravity, the roll angle φ and the respective velocities $\dot{y}$, $\dot{z}$ and $\dot{\varphi}$. The control inputs u_1 and u_2 are respectively the vertical thrust force and the rolling moment. The rolling moment u_2 generates also a lateral force (because the lift forces are not perpendicular to the wings) and ϵ_{PVTOL} is the coupling coefficient. Finally g is the gravity acceleration. In Figure 3.1 the PVTOL aircraft with the reference system and the inputs is shown.

In [32] the PVTOL was shown to be input-output linearizable when $\epsilon_{\text{PVTOL}} = 0$, while in [46] it was shown that, when ϵ_{PVTOL} is not zero, it is possible to find suitable outputs (flat outputs) such that the system can be feedback linearized by means of dynamic extension. Using as flat outputs

$$\begin{aligned} y_{\text{f}} &= y + \epsilon_{\text{PVTOL}} \sin(\varphi) \\ z_{\text{f}} &= z + \epsilon_{\text{PVTOL}} \cos(\varphi) \end{aligned}$$

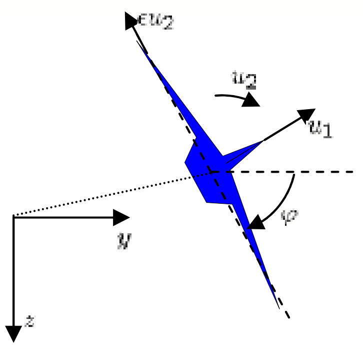


Figure 3.1: PVTOL aircraft.

after some straightforward calculations one can easily show that:

$$\begin{aligned}
\dot{y}_{\mathrm{f}} &= \dot{y} + \epsilon_{\mathrm{PVTOL}}\, \dot{\varphi} \cos(\varphi) \\
\ddot{y}_{\mathrm{f}} &= \tilde{u}_1 \sin(\varphi) \\
y_{\mathrm{f}}^{(3)} &= \dot{\tilde{u}}_1 \sin(\varphi) + \tilde{u}_1 \dot{\varphi} \cos(\varphi) \\
\dot{z}_{\mathrm{f}} &= \dot{z} - \epsilon_{\mathrm{PVTOL}}\, \dot{\varphi} \sin(\varphi) \\
\ddot{z}_{\mathrm{f}} &= -\tilde{u}_1 \cos(\varphi) \\
z_{\mathrm{f}}^{(3)} &= -\dot{\tilde{u}}_1 \cos(\varphi) + \tilde{u}_1 \dot{\varphi} \sin(\varphi),
\end{aligned}$$

where $\tilde{u}_1 = u_1 - \epsilon_{\mathrm{PVTOL}}\, \dot{\varphi}^2$. Clearly, for all $\dot{\varphi}$ and u_1 such that $\tilde{u}_1 \neq 0$, the system is feedback linearizable and in fact equivalent to the two dimensional forth order integrator

$$\begin{aligned}
y_{\mathrm{f}}^{(4)} &= v_1 \\
z_{\mathrm{f}}^{(4)} &= v_2,
\end{aligned}$$

with suitable expressions for v_1 and v_2.

Remark 3.1 *For $\epsilon_{\mathrm{PVTOL}} = 0$ the condition $\tilde{u}_1 \neq 0$ reduces to $u_1 \neq 0$. This means that if u_1 is positive (physical constrain) and bounded away from zero the system can be always feedback linearized despite of the value of the states.* □

The PVTOL has relative degree $r = [2, 2]$. Posing

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \sin\varphi & -\cos\varphi \\ \frac{-\cos\varphi}{\epsilon_{\mathrm{PVTOL}}} & \frac{-\sin\varphi}{\epsilon_{\mathrm{PVTOL}}} \end{bmatrix} \left(\begin{bmatrix} 0 \\ -g \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \right)$$

the dynamics of the system become

$$\begin{aligned} \ddot{y} &= v_1 \\ \ddot{z} &= v_2 \\ \ddot{\varphi} &= \tfrac{(g-v_2)}{\epsilon_{\mathrm{PVTOL}}}\sin\varphi - \tfrac{v_1}{\epsilon_{\mathrm{PVTOL}}}\cos\varphi. \end{aligned} \tag{3.2}$$

As expected, the output dynamics are linear, however the zero dynamics is unstable. Equation (3.2) is the dynamics of a *Driven Pushed Pendulum.* In this sense the PVTOL can be seen as a general case of many other mechanical systems sharing a pendulum-like dynamics.

An important role in the study of the trajectory manifold of the PVTOL is played by the "quasi trajectory" that (with some abuse of notation) we call *quasi-static trajectory.* It is defined as:

$$\tan\varphi_{qs}(t) = \frac{\ddot{y}(t)}{(g - \ddot{z}(t))}$$

and it is a curve built pretending that at each instant the roll angle assumes the equilibrium value as $\ddot{y}(t)$ and $\ddot{z}(t)$ were constant. It is worth noting that the quasi static roll trajectory is exactly the roll trajectory for the model with $\epsilon_{\mathrm{PVTOL}} = 0$. This provides a further motivation, in the next section, to search a roll trajectory "close" to the quasi static approximation.

3.3 Trajectory exploration: the unconstrained PVTOL

In this section we study the trajectory manifold of the unconstrained PVTOL.

3.3.1 Dichotomy and existence of a bounded roll trajectory

The objective is to explore the trajectory manifold of the *unconstrained* PVTOL using the y and z trajectories of the center of gravity as "parameters". In other words we can summarize our objective as follows:

Problem: Given $y_d(\cdot)$ and $z_d(\cdot)$ "sufficiently smooth" trajectories of the center of gravity, we want to find a "bounded" roll trajectory $\varphi(\cdot)$ satisfying

$$\ddot{\varphi} = \tfrac{1}{\epsilon_{\mathrm{PVTOL}}}(g - \ddot{z}_d)\sin\varphi - \tfrac{1}{\epsilon}\ddot{y}_d\cos\varphi. \tag{3.3}$$

We have to formalize what we mean for "sufficiently smooth" and "bounded" trajectories. First of all we assume that $y_d(\cdot)$ and $z_d(\cdot)$ are C^2 trajectories. Moreover we define

$$\alpha^2_{\text{PVTOL}}(t) = \ddot{y}^2_d + (\ddot{z}_d - g)^2, \tag{3.4}$$

and we assume

$$0 < a_{\min} \leq \alpha_{\text{PVTOL}}(t) \leq a_{\max}. \tag{3.5}$$

Now we rewrite the roll dynamics (3.3) in the form

$$\ddot{\varphi} = \tfrac{1}{\epsilon_{\text{PVTOL}}} \alpha_{\text{PVTOL}}(t) \sin(\varphi - \varphi_{qs}(t)). \tag{3.6}$$

where $\varphi_{qs}(t)$ is the quasi-static roll angle and satisfies

$$\tan \varphi_{qs}(t) = \tfrac{\ddot{y}_d(t)}{(g - \ddot{z}_d(t))}. \tag{3.7}$$

In Figure 3.2 the space of the admissible accelerations is shown with the quasi-static roll angle. Observe that the quasi static angle is just the angle for which the pendulum is aligned with the direction of the resultant acceleration so that at that instant the rolling moment is zero. Hence, the task is to find a bounded roll trajectory in the sense that the difference

$$\theta(t) = \varphi(t) - \varphi_{qs}(t) \tag{3.8}$$

is bounded. Given a $\mathcal{C}^1$ output trajectory (of the center of gravity) on the

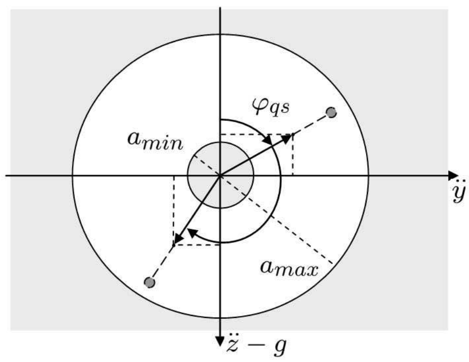


Figure 3.2: Acceleration configuration and quasi-static roll angle

infinite time interval $\mathbb{R}$, satisfying

$$0 < \alpha_{\text{PVTOL}}(t) < a_{\max}, \ \forall t \in \mathbb{R},$$

there exists a bounded roll trajectory consistent with the output one.

Remark 3.2 (Acceleration bound in terms of u_1) *It is worth noting that* $|u_1(t)| = \alpha_{\text{PVTOL}}(t)$*, therefore, starting with* $u_1(0) > 0$*, the above condition is equivalent to* $0 < u_1(t) < a_{max}$*,* $\forall t \in \mathbb{R}$*.* □

The proof of existence is based on the presence of a dichotomy in the linearization of the pendulum dynamics about the vertical position. The bounded trajectory is proven to exist as a fixed point of a contraction mapping [31].

The roll dynamics in (3.2) can be rewritten in terms of the error from the *quasi-static* angle, $\theta = \varphi - \varphi_{qs}$, as

$$\begin{aligned}\ddot{\theta} &= \alpha^2_{\text{PVTOL}}(t)\sin(\theta) + \ddot{\varphi}_{qs}(t)\\ &= \alpha^2_{\text{PVTOL}}(t)\theta - \alpha^2_{\text{PVTOL}}(t)\Big(\theta - \sin\theta + \tfrac{\ddot{\varphi}_{qs}(t)}{\alpha^2_{\text{PVTOL}}}\Big)\end{aligned} \tag{3.9}$$

where $\alpha^2_{\text{PVTOL}}(t) = \sqrt{(g - v_2(t))^2 + v_1^2(t)}/\epsilon_{\text{PVTOL}}$ and in the second line we have rewritten the dynamics in terms of its linearization about the equilibrium $\theta = 0$. If we consider the linear time-varying system driven by a bounded external input

$$\ddot{\gamma} = \alpha^2_{\text{PVTOL}}(t)\gamma - \alpha^2_{\text{PVTOL}}(t)\mu(t),$$

it can be proven [31] that the undriven system admits an exponential dichotomy and, therefore that, working in a noncausal fashion, for any bounded input $\mu(\cdot)$ a bounded solution $\gamma(\cdot)$ exists. The linear map $\mu(\cdot) \to \gamma(\cdot)$ is denoted by $\mathcal{A}$.
Defining the nonlinear operator $\mathcal{N}$

$$\theta \to \mathcal{A}\left[\theta - \sin\theta + \ddot{\varphi}_{qs}(t)/\alpha^2_{\text{PVTOL}}(\cdot)\right] =: \mathcal{N}\left[\theta(\cdot)\right],$$

it is easy to see that a bounded curve $\theta(\cdot)$ is a solution of (3.9) if and only if it is a fixed point of $\mathcal{N}$, i.e. $\theta(\cdot) = \mathcal{N}[\theta(\cdot)]$. The following theorem is proven in [31].

Theorem 3.3 (Existence of a bounded roll trajectory) *If*

$$\|\mathcal{A}\left[\ddot{\varphi}_{qs}(\cdot)/\alpha^2_{\text{PVTOL}}(\cdot)\right]\| < 1$$

then there is a $\delta < \pi/2$ *such that* $\mathcal{N}$ *is a contraction on the invariant set* $\bar{B}_\delta$*. The unique fixed point* $\theta(\cdot)$ *of* $\mathcal{N}$ *in* $\bar{B}_\delta$ *is a bounded trajectory of* (3.9) *so that* $\varphi(\cdot) = \varphi_{qs}(\cdot) + \theta(\cdot)$ *is a bounded roll trajectory of* (3.2)*.* □

We refer the reader to [31] for the proof and [53] for a detailed treatment on the PVTOL case.

3.3.2 Bounded roll trajectory computation

In order to compute a bounded roll trajectory on a finite horizon $[0, T]$, we use the dynamics embedding technique introduced in [30] and described in Chapter 2, based on the nonlinear projection operator Newton method. The method applies to the PVTOL by embedding the original roll dynamics in the driven system

$$\ddot{\varphi} = \alpha^2_{\text{PVTOL}}(t)\sin(\varphi - \varphi_{qs}(t)) + u_{ext}, \tag{3.10}$$

where u_{ext} is the fictitious input used to drive the system along any desired feasible trajectory. If the accelerations of the center of gravity vary slowly, we can imagine the roll trajectory to be close to the quasi-static one. Hence, we can use the quasi-static as an initial guess to find the real trajectory. If we rewrite (3.10) in state space form as $\dot{x}_\phi = f_\phi(t, x_\phi, u_{ext})$, where $x_\phi = (\varphi, \dot{\varphi})$ and $x_{qs} = (\varphi_{qs}, \dot{\varphi}_{qs})$, we may pose the following optimization problem:

$$\begin{aligned}
&\text{minimize}\\
&h(x_\phi, u_{\text{ext}}) = \tfrac{1}{2}\int_0^T \|x_\phi(\tau) - x_{qs}(\tau)\|^2_Q + \rho|u_{\text{ext}}(\tau)|^2 d\tau\\
&\qquad\qquad\quad + \tfrac{1}{2}\|x_\phi(T) - x_{qs}(T)\|^2_{P_1}\\
&\text{subject to } \dot{x}_\phi = f_\phi(t, x_\phi, u_{\text{ext}}).
\end{aligned}$$

where Q, ρ and P_1 are weighting parameters. Using a high weight ρ for the input, we may obtain a trajectory arbitrarily close to the bounded roll trajectory we are looking for. The optimization problem is solved by using the projection operator based Newton method described in the next session.

3.3.3 Simulations

We show two examples of maneuvers. We used a value of 0.3 for ϵ_{PVTOL} and we normalized gravity to one for this simulation. We performed a circular trajectory in the $y - z$ plane, i.e. we asked the PVTOL to track the trajectories $y_d(t) = -z_{max}/\omega_0^2 \sin(\omega_0 t)$ and $z_d(t) = -z_{max}/\omega_0^2(1 - \cos(\omega_0 t))$ where $\omega_0 = 2\pi 1.5/10$. The vertical acceleration was set to different values in the two simulations. In particular, in the first case we chose $|z_{max}| = 0.9g$, so that the resulting vertical acceleration $g - \ddot{z}_d(t)$ remains always positive. As shown in Figure 3.3 the maneuver results in an upright roll trajectory. The red dashed lines represent the resulting accelerations. Observe that the quasi static model (green) is aligned at each instant along the acceleration direction.

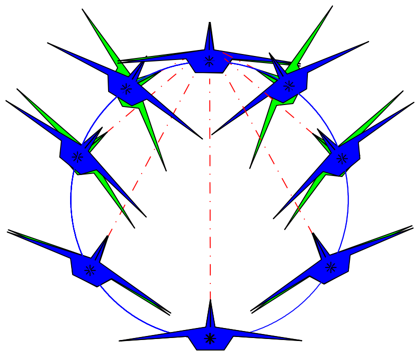
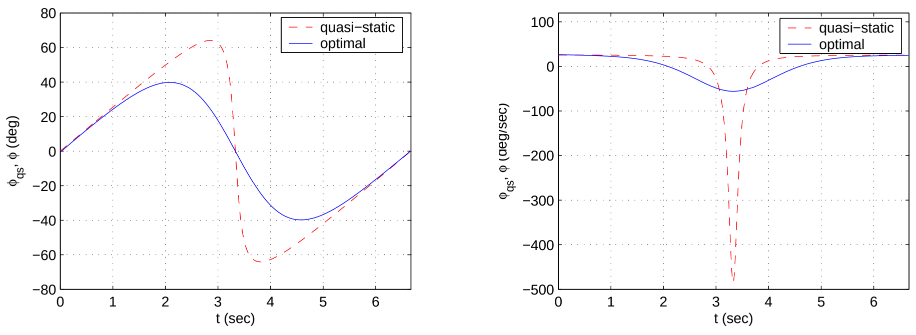


Figure 3.3: Upright trajectory: optimal (blue) and quasi-static (green)

In Figure 3.4a and Figure 3.4b the roll and roll rate optimal trajectories are compared with the quasi-static ones. The optimal trajectories display a degree of anticipation and are smoother than the quasi-static due to the filtering action of the dynamics.



(a) roll (b) roll rate

Figure 3.4: Quasi-static versus optimal roll and roll-rate trajectories (upright trajectory)

In the second scenario we set $|z_{max}| = 1.2g$. In this way the global vertical acceleration $g - \ddot{z}_d(t)$ changes sign. An upright roll trajectory does not exists and a barrel roll occurs Figure 3.5. In Figure 3.6a and in Figure 3.6b the optimal and quasi static roll and roll rate trajectories are compared as in the previous case.

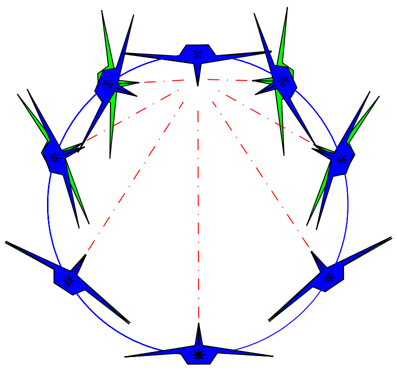
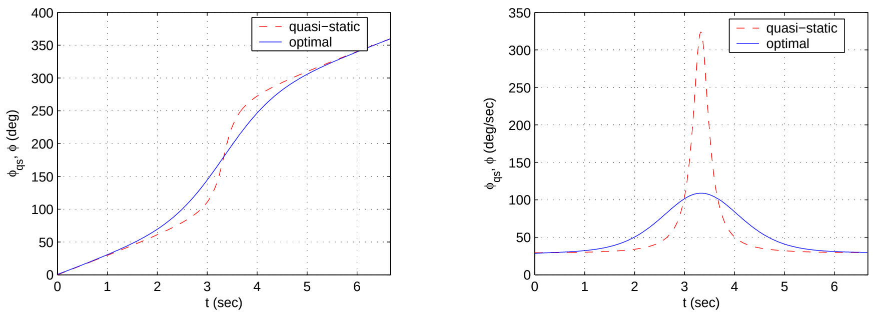


Figure 3.5: Barrel roll trajectory: optimal (blue) and quasi-static (green)



(a) roll (b) roll rate

Figure 3.6: Quasi-static versus optimal roll and roll-rate trajectories (barrel roll trajectory)

3.4 Computing feasible trajectories

In this section we define the operating conditions for the PVTOL and then apply the strategy described in Chapter 2 to find feasible trajectories.

We start by assigning the task for the constrained PVTOL. As seen in the previous chapter, we need to specify a desired output curve and a task control system for the task and then a set of operating conditions. The objective output of the PVTOL consists of the coordinates of the gravity center. The task control system that we use for the PVTOL is the model with $\epsilon_{\text{PVTOL}} = 0$. We call it PVTOL$_0$ to distinguish from the nominal model, that in this section we refer to as PVTOL$_\epsilon$. The PVTOL$_0$ is flat with respect to its output. A bounded roll trajectory may be easily obtained by (3.7). That is, the quasi-static trajectory is a roll trajectory of PVTOL$_0$.

We set the following operating conditions on the control inputs, parameterized by $\rho \geq 1$,

$$\left(\frac{u_1 - \rho g}{\rho g}(1 + \frac{1}{\rho^2})\right)^2 \leq 1$$

and

$$\left(\frac{u_2}{\rho u_{2\max}}\right)^2 \leq 1$$

where the nominal conditions are obtained for $\rho = 1$. Notice that starting with $u_1(0) > 0$ the physical bound of a positive thrust is also ensured.

Using flatness of PVTOL$_0$ with respect to its output, we may easily derive constraints on the desired output trajectory, so that it satisfies the operating conditions. The constraints may be imposed by using Remark 3.2 together with Equation (3.4) for u_1 and Equation (3.7) for u_2.

In many applications, trying to impose such constraints may be hard or may result in too conservative desired trajectories. Using the barrier function based exploration strategy even for the PVTOL$_0$, we may obtain a desired trajectory that satisfies the operating conditions and is really close to a desired output trajectory chosen without imposing the constraints.

Remark 3.4 (Operating region for the PVTOL) *It is worth noting that in the input region described by the operating conditions, the PVTOL$_0$ is feedback linearizable ($u_1 > 0$). For ϵ_{PVTOL} not zero the constraints may be suitably modified so that the same holds for the PVTOL$_\epsilon$. In the next chapter we will show that, if we consider a compact subset of the state-input space (such that* *the input portion satisfies the operating conditions), any trajectory of the system staying in it is uniformly linearly controllable. That is, the linearization of the system about any trajectory is uniformly controllable. We will call this compact set* operating region. *Interesting features of this region are analyzed in the next chapter.* □

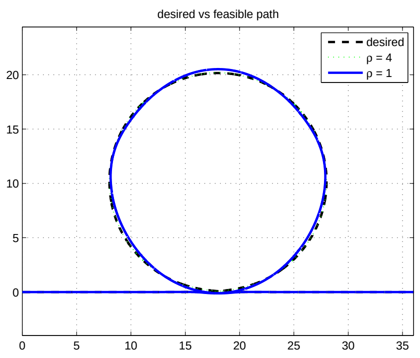
In order to explore feasible trajectories we follow two different approaches based on the homotopy and dynamics embedding methods described in Chapter 2. In the first approach we start computing a feasible trajectory for PVTOL$_0$ and then we use the homotopy method. Differently from what described in the previous chapter the homotopy method is applied by using the modified cost with the barrier function, so that the trajectories computed at each step are all feasible. The second approach to solve the problem is based on the fact that we can find a bounded roll trajectory of the unconstrained PVTOL$_\epsilon$ whose output is exactly the desired one. The strategy is the following. Given the desired output curve (that we recall is a trajectory of PVTOL$_0$), we find an unconstrained trajectory of PVTOL$_\epsilon$ by means of the dynamics embedding technique described in the previous section. Then, we use the barrier function based strategy to find a feasible trajectory close to the unconstrained one.

3.5 Simulations

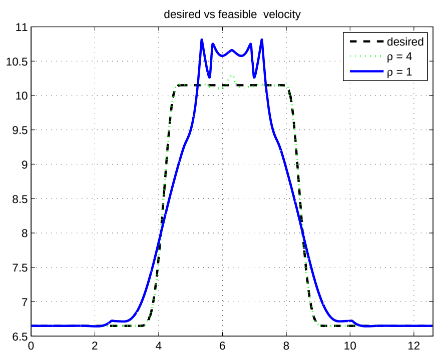
We present the simulation results for the PVTOL with $\epsilon_{\text{PVTOL}} = 0.05$. We perform a barrel roll subject to the bounds on u_1 and u_2 defined above. We used the approach based on dynamics embedding. The desired path is the one depicted in Figure 3.7a with a dashed line. A velocity profile is also assigned on the path, Figure 3.7b. The desired velocity profile is constant to $v_0 = 6.65 m/s$ in the first and last flat portions of the path and goes smoothly to $v_0 = 10.15 m/s$ in the central loop. Figure 3.7 compares the desired path and velocity with the feasible ones respectively computed for $\rho = 4$, an intermediate value for which the constraints are only slightly violated, and $\rho = 1$ that provides the specified constraints. In Figure 3.8 the position errors are shown. Finally, in Figure 3.9 desired and feasible inputs are compared. The optimization was performed by iterating the barrier function method starting with $\epsilon_c = 1$ up to $\epsilon_c = 0.01$. The maximum error on $y(\cdot)$ and $z(\cdot)$ is found to be $0.5m$. The result is quite surprising considering the tight limit on the thrust and roll moment. As it can be seen in Figure 3.9 the control tends to hit the boundary for a larger interval of time than the one where the constraints are violated (thus working in a noncausal fashion) in order to compensate the missing availability of input effort in those regions. Also, since we penalized the difference between feasible and desired controls very little, they result to be quite different.

3.6 Discussion

In this chapter we have implemented for the PVTOL aircraft the exploration strategy based on barrier function optimization, described in Chapter 2. A future direction of research is the application of the strategy to a rigid aircraft model.



(a) path



(b) velocity

Figure 3.7: Desired vs feasible path and velocity.

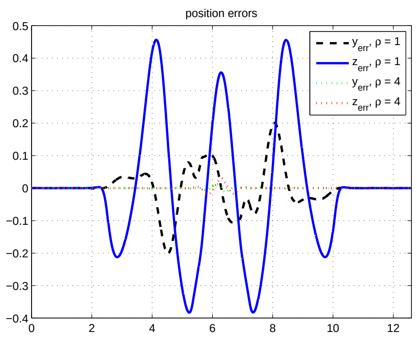
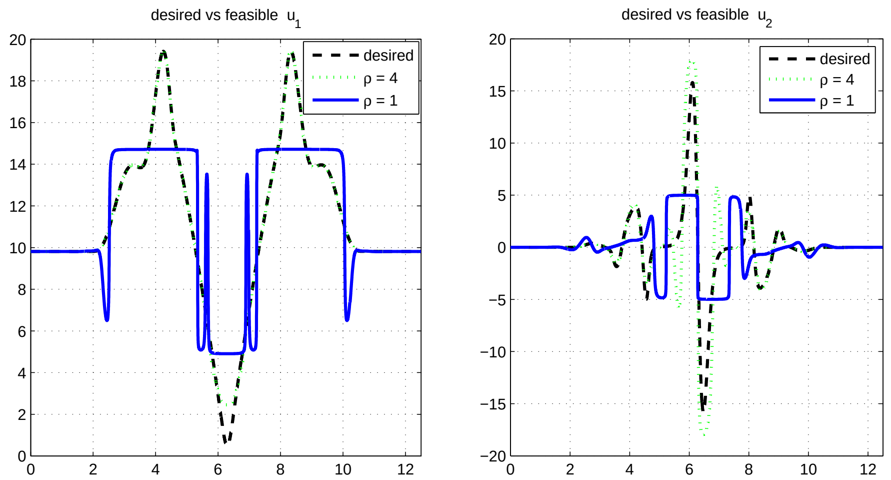


Figure 3.8: Position errors



(a) u_1 (b) u_2

Figure 3.9: Desired vs feasible u_1 and u_2.

Chapter 4

Operating region

In this chapter we want to give some preliminary answers to the following problem. Given a nonlinear control system, can we find a subset of the state-input space, such that any trajectory remaining in it is uniformly linearly controllable? That is, we ask the linearization of the system about any of these trajectories to be uniformly controllable. We believe this problem is of great engineering interest. In fact, if such a region exists, the barrier functional optimization strategy may be used in this region to find exponentially stabilizable trajectories, thus obtaining an effective receding horizon scheme. We characterize an operating region for feedback linearizable systems, and for control-affine systems driven by sampled controls. We also prove that state trajectories generated by sampled controls converge uniformly to actual state trajectories. This suggests us that the operating region for sampled-control trajectories is a good candidate operating region for any bounded trajectory.

4.1 Introduction

In Chapter 2 we have introduced an optimal control based strategy to find feasible trajectories, that is, trajectories satisfying bounds on states and/or inputs. We called such bounds operating conditions to underline their engineering meaning. We have also discussed how to apply this strategy in a receding horizon scheme to perform trajectory tracking.

In order to apply the receding horizon scheme, we need the desired feasible trajectory to be exponentially stabilizable. A sufficient condition for that is uniform linear controllability (ULC) about the feasible trajectory (that is the time-varying linearization is uniformly linearly controllable). This means that every time we compute a new desired feasible trajectory, we must verify one of such properties before applying the receding horizon control. It is clear that this is a strong concern, especially if we would implement a hierarchical strategy that includes both the exploration and the tracking strategies.

Another reason to ask for uniform linear controllability is to apply the exploration strategy in two point boundary value problems. This could provide, for example, a useful tool to perform path planning.

In this chapter we study whether we can provide sufficient conditions to ensure uniform linear controllability on a subset of the state-input space. If this and other technical conditions are satisfied, we call such subset an *operating region*. Two excellent surveys for nonlinear and linear controllability are [34] and [33]. A similar problem was studied by Sontag in [56]. He characterized the set of nonsingular controls of a real-analytic nonlinear system. A control is said nonsingular if it produces a trajectory along which the linearized system is controllable. He showed that if the system is strongly accessible, the set of smooth universal[1] nonsingular controls is generic in $\mathcal{C}^\infty$. The proof of this result is based on results proven by Sussmann in [60]. In [57] a numerical technique for path planning of drift-less systems, relying on the existence of nonsingular controls, was developed. In [11], Coron used perturbations of nonsingular controls to asymptotically stabilize nonlinear control systems.

The study on this problem is still at a preliminary stage. We provide some prototype results that should help us, in future work, to provide sufficient conditions to characterize an operating region for control affine systems driven by measurable bounded inputs. The contribution of this chapter is three-fold. First, we prove the existence of an operating region for feedback linearizable systems. In particular, we prove that any compact subset of the state-input space where the system is feedback linearizable, with the addiction of technical assumptions, is in fact an operating region. Second, we provide sufficient conditions to characterize an operating region for control affine systems driven by piecewise constant inputs. Third, we show that state trajectories generated by sampled controls converge uniformly to actual state trajectories. This suggests us that the operating region for sampled-control trajectories is a good candidate operating region for any bounded trajectory.

[1]A control is called a universal nonsingular control if it is nonsingular for every state.

4.2 Uniform linear controllability of nonlinear systems

In this section we introduce some notions regarding controllability of nonlinear systems and their time-varying linearization about given trajectories. The main concept introduced in the section is the *linear controllability* of nonlinear systems. It will play an important role in defining and characterizing the notion of operating region.

We begin by recalling some basic notions. We consider nonlinear control systems, with states lying in $\mathbb{R}^n$ and inputs in $\mathbb{R}^m$, of the form

$$\dot{x}(t) = f(x(t), u(t)), \tag{4.1}$$

for all $t \geq 0$, where f is a $\mathcal{C}^r$ mapping of the states and inputs, $r \in \{1, \dots, +\infty\}$. We recall that a trajectory of system (4.1) is a *bounded* curve that satisfies the differential equation, that is $\xi = (\alpha(\cdot), \mu(\cdot))$, $\mu(\cdot)$ bounded (with respect to the L_∞ norm), and $\dot{\alpha}(t) = f(\alpha(t), \mu(t))$ for all $t \geq 0$.

Given a trajectory $\xi = (\alpha(\cdot), \mu(\cdot))$ of a control system Σ as in (4.1), the differential equation

$$\dot{z} = A(\xi(t))z + B(\xi(t))v, \tag{4.2}$$

where $A(\xi(t)) = D_x f(\alpha(t), \mu(t))$ and $B(\xi(t)) = D_u f(\alpha(t), \mu(t))$, is called *variational equation* or *(time-varying) linearization of* Σ *about* ξ, and $\zeta = (z(\cdot), v(\cdot))$ is called the *variation* of ξ.

We are now ready to introduce the notion of (uniform) linear controllability. Roughly speaking we say that a nonlinear system is (uniformly) linearly controllable about a trajectory if the time-varying linearization of the system about the trajectory is (uniformly) controllable. Formally, we have the following definition.

Definition 4.1 (Uniform linear controllability (ULC)) *Given the time invariant nonlinear system* Σ

$$\dot{x} = f(x, u),$$

as in (4.1) *and a trajectory* ξ *of* Σ*, the system is "*(uniformly) linearly controllable about $\xi(\cdot)$*" (or* ξ *is a (uniformly) linearly controllable trajectory of* Σ*), if the time-varying linearization*

$$\dot{z} = A(\xi(t))z + B(\xi(t))v,$$

with $A(\xi(t)) = D_x f(\alpha(t), \mu(t))$ and $B(\xi(t)) = D_u f(\alpha(t), \mu(t))$, is (uniformly) controllable. If the property holds for any $\xi = (\alpha(\cdot), \mu(\cdot))$, such that $(\alpha(t), \mu(t)) \in \mathcal{XU} \subseteq \mathbb{R}^n \times \mathbb{R}^m$, then the system is said (uniformly) linearly controllable on $\mathcal{XU}$. $\square$

We recall the notion of controllability and uniform controllability for a linear time-varying system, and some useful properties.

Definition 4.2 (Controllability of LTV systems) *A linear time-varying system*

$$\dot{z} = A(t)z + B(t)v, \tag{4.3}$$

$A(t)$ and $B(t)$ bounded and continuous, is said to be "(completely) controllable on $[t_0, t_1]$"*, $0 \leq t_0 < t_1 < \infty$, if for each z_0 and z_1 in $\mathbb{R}^n$, there exists a bounded measurable control $v(\cdot)$ that drives the system from z_0 at time t_0 to z_1 at time t_1.* $\square$

The following proposition states some interesting properties of controllable linear systems.

Proposition 4.3 (Equivalent conditions for controllability) *Let a linear time-varying system as in Definition 4.2 be given. The following are equivalent:*

(i) *the linear time-varying system is (completely) controllable on $[t_0, t_1]$;*

(ii) *the* reachability grammian

$$W_R(t_0, t_1) = \int_{t_0}^{t_1} \Phi(t_1, \tau) B(\tau) B^T(\tau) \Phi^T(t_1, \tau) d\tau$$

is positive definite;

(iii) *the grammian*

$$W_0(t_0, t_1) = \int_{t_0}^{t_1} \Phi(t_0, \tau) B(\tau) B^T(\tau) \Phi^T(t_0, \tau) d\tau$$

is positive definite;

(iv) *the rows of $\Phi(t_0, \tau)B(\tau)$ are linearly independent on $[t_0, t_1]$.*

$\square$

Remark 4.4 (Controllability is preserved on larger intervals) *If the system is (completely) controllable on $[t_0, t_1]$, then it is (completely) controllable on every $[t_2, t_3] \supset [t_0, t_1]$. This follows easily by the grammian condition.* $\square$

Remark 4.5 (Meaning of "completely" about controllability) *The use of "completely" in the controllability definition is archaic — it will be dropped in the sequel. It was used, in the past, to underline that the property holds for every pair of states at every initial and final time. If it is not so we simply say that the system is not controllable.* $\square$

An important property of the controllability grammian W_R is that, it can be obtained as the state of the dynamic system

$$\dot{Q}(t) = A(t)Q(t) + Q(t)A(t) + B(t)B^T(t), \qquad Q(0) = 0, \tag{4.4}$$

where $W_R(t, 0) = Q(t)$.

With the notion of controllability grammian in hand, we may introduce the definition of uniform controllability of linear time-varying systems. Informally, a linear system is said uniformly controllable on the interval $[t_0, t_1[$ if there is a $\delta > 0$ such that the norm of the grammian is bounded away from zero on every sub-interval of $[t_0, t_1[$ of length δ. Formally, we have the following definition.

Definition 4.6 (Uniform controllability of LTV systems) *A linear time-varying system* (4.3)*, with $A(t)$ and $B(t)$ bounded and continuous, is said to be* "uniformly controllable on $[t_0, t_1[$", $0 \le t_0 < t_1 \le +\infty$*, if there exist $\delta > 0$ and $k_\delta > 0$ such that, for all $t \in [t_0, t_1 - \delta[$,*

$$\int_t^{t+\delta} \Phi(t, \tau)B(\tau)B^T(\tau)\Phi^T(t, \tau)d\tau \ge k_\delta I.$$

Furthermore it is said " δ_c-uniformly controllable on $[t_0, t_1[$" *if it is uniformly controllable with $\delta = \delta_c$.* $\square$

Remark 4.7 (Uniform controllability with larger δ) *Clearly, if the system is δ_1-uniformly controllable, then it is δ_2-uniformly controllable for any $\delta_2 \ge \delta_1$.* $\square$

The following lemma is due to Silverman and Anderson [55].

Lemma 4.8 (Condition for uniform controllability of LTV systems) *A linear time-varying system* (4.3)*, with $A(t)$ and $B(t)$ bounded, is uniformly (completely) controllable if and only if there exist $\delta > 0$ and k_δ such that for every state $\bar{z} \in \mathbb{R}^n$ and for any time t, there exists an input v defined on $]t-\delta, t[$ such that, if $z(t-\delta_c) = 0$, then $z(t) = \bar{z}$ and $\|v(\tau)\| \leq \gamma_\delta \|\bar{z}\|$ for all $\tau \in]t-\delta, t[$.* □

A stronger notion than the simple controllability, introduced above, is the impulsive controllability. It is a point-wise condition that involves the derivatives of $A(\cdot)$ and $B(\cdot)$. Therefore it can be used only if these are sufficiently differentiable.

Definition 4.9 (Impulsive controllability of LTV systems) *A linear time-varying system is said to be* "impulsively controllable at t" *iff, defining the operator $\mathcal{A} = \frac{d}{dt} - A(\cdot)$,*

$$\mathit{rank}\{B(t), (\mathcal{A}^k B(\cdot))(t), k \geq 1\} = n.$$

□

Remark 4.10 (Impulsive and uniform controllability) *Even for impulsive controllability we can ask whether the property is uniform over some interval (possibly of infinite length). In general impulsive controllability neither implies or is implied by uniform controllability.* □

4.3 Definition and characterization of operating region

In this section we introduce the definition of operating region and provide sufficient conditions to characterize it for feedback linearizable systems and for control affine systems driven by piecewise constant inputs.

Here is an informal description of what we shall refer to as operating region.

> *Operating region [Informal description]* An operating region is a portion of the state-input space with the property that every admissible trajectory laying in it satisfies some structural properties of the system. That is, no actuator saturation occurs, the model

used to represent the system is valid, dynamics constraints are satisfied and, furthermore, the system is uniformly linearly controllable on it.

An admissible trajectory is simply a trajectory of the system $\xi = (\alpha(\cdot), \mu(\cdot))$, whose input $\mu(\cdot)$ belongs to a class of admissible functions $\mathcal{F}_u$. We are interested in the class of bounded measurable inputs. A useful class in characterizing an operating region will be the one of piecewise constant inputs.

In the following we define the operating region more formally.

Definition 4.11 (Operating region) *Let Σ be a control system as in* (4.1) *and $\mathcal{F}_u \subset L_\infty$ the space of admissible input functions. An operating region for Σ with respect to the admissible input set $\mathcal{F}_u$ is an open simply connected set $\mathcal{XU}$ such that*

- *its closure $\overline{\mathcal{XU}} \subset \mathbb{R}^n \times \mathbb{R}^m$ is compact;*
- *$\mathcal{U} = \{u \in \mathbb{R}^m \mid (x, u) \in \mathcal{XU}$ for every fixed $x\}$ is convex;*
- *$\mathcal{XU} \subset \mathcal{XU}_{OC}$, where $\mathcal{XU}_{OC} \subset \mathbb{R}^n \times \mathbb{R}^m$ is an open simply connected set of state-input constraints (operating conditions);*
- *for any trajectory $\xi = (\alpha(\cdot), \mu(\cdot))$ of Σ, such that $(\alpha(t), \mu(t)) \in \mathcal{XU}$ for all $t \geq 0$, then Σ is uniformly linearly controllable about ξ.*

□

Let us comment on the definition. The assumptions $\mathcal{XU}$ simply connected and $\overline{\mathcal{XU}}$ compact are technical assumptions. We ask for convexity of $\mathcal{U}$ because we want to use the operating region as the domain of an optimal control based strategy to find feasible trajectories of the system. In such setting convexity of the input portion plays a key role in the existence of optimal solutions. The set $\mathcal{XU}_{OC}$ has an engineering meaning and is basically the one introduced in Chapter 2. It is a portion of the state-input space that is characterized in the design of the system or in the definition of its model representation. It is a region where physical or dynamics constraints are satisfied. The third ingredient is the one we are more interested on in this chapter for the reasons discussed in Section 4.1.

4.3.1 Uniform linear controllability for feedback linearizable systems

Before stating our first result, we recall the notion of feedback linearizable systems on an open subset of the state space.

We consider a control affine system

$$\dot{x}(t) = f_0(x(t)) + g(x(t))u(t), \tag{4.5}$$

where $g(x) = [g_1(x) \dots g_m(x)]$ and the same assumptions as in (4.1) hold.

In order to state notion of feedback linearizable systems on an open set, we define the *state space exact linearization problem* as stated in [35].

Definition 4.12 (State space exact linearization problem) *Given a set of vector fields $f_0(x)$ and $g_1(x), \dots, g_m(x)$ and a initial state x_0, find (if possible) a neighborhood X_0 of x_0, a pair of feedback functions $\alpha(x)$ and $\beta(x)$ defined on X_0, a coordinate transformation $y = \Psi(x)$ also defined on X_0, a matrix $A \in \mathbb{R}^{n\times n}$ and a matrix $B \in \mathbb{R}^n \times m$, such that*

$$[D\Psi(x)(f_0(x) + g(x)\alpha(x))]_{x=\Psi^{-1}(y)} = Ay \tag{4.6}$$

$$[D\Psi(x)g(x)\beta(x)]_{x=\Psi^{-1}(y)} = B \tag{4.7}$$

and

$$rank(B \ AB \dots \ A^{n-1}B) = n. \tag{4.8}$$

$\square$

Let $X \subset \mathbb{R}^n$ be a simply connected open set. We say that a system Σ as in 4.5 is feedback linearizable on X, if the state-space exact linearization problem is solvable for any $x \in X$.

In the following proposition we prove uniform linear controllability of feedback linearizable systems over a compact subset of the state-input space.

Proposition 4.13 (Feedback linearization and ULC) *Suppose that the control affine system Σ as in (4.5) is feedback linearizable on the simply connected open set $X_f \subset \mathbb{R}^n$. Let $\mathcal{XU} = X \times U$, where X and U are open, bounded and simply connected sets and the closure of $\mathcal{XU}$, $\overline{\mathcal{XU}} = \overline{X} \times \overline{U} \subset X_f \times \mathbb{R}^m$ and is compact. Then, Σ is uniformly linearly controllable on $\mathcal{XU}$. That is, every trajectory remaining in $\mathcal{XU}$ for all $t \geq 0$, with $u(\cdot)$ bounded and measurable, is δ-uniformly linearly controllable for any $\delta > 0$.*

Proof: We start proving linear controllability and then we prove that it is uniform.

Since Σ is feedback linearizable on $X_f \supset \overline{X}$, there exist a pair of feedback functions $\alpha(x)$ and $\beta(x)$ defined on $\overline{X}$, with $u = \alpha(x) + \beta(x)\omega$, $\omega \in \mathbb{R}^m$, a coordinate transformation $y = \Psi(x)$ also defined on $\overline{X}$, and a pair of constant matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$, such that, for $x \in \overline{X}$, (4.6), (4.7) and (4.8) hold.

That is, the system

$$\dot{y} = Ay + B\omega, \tag{4.9}$$

with $y \in \mathbb{R}^n$ and $\omega \in \mathbb{R}^m$, is a (uniformly) controllable linear time-invariant system.

This implies that there exist $W(x)$ and $b(x)$ defined on $\overline{X}$ such that

$$\omega = b(x) + W(x)u, \tag{4.10}$$

where $\alpha(x) = -W^{-1}(x)b(x)$ and $\beta(x) = W^{-1}(x)$.

Let $\mathcal{F}_u$ be the set of (essentially) bounded input functions. If we would prove (nonlinear) controllability, we should prove that the mapping $\mathcal{R} : \mathcal{F}_u \to \mathbb{R}^n$, defined as

$$u(\cdot) \mapsto \phi(T; x_0, u(\cdot))$$

is surjective. In order to prove linear controllability, we need to show that the differential of the mapping $\mathcal{R}$ is surjective.

We can consider the mapping $\mathcal{R}$ as the composition of three different mappings. First, we consider the mapping $\mathcal{B} : \mathbb{R}^n \times \mathcal{F}_u \to \mathbb{R}^n \times \mathcal{F}_u$ defined as

$$(x_0, u(\cdot)) \mapsto (y_0, \omega(\cdot))$$

and given by

$$(y_0, \omega(\cdot)) = (\Psi(x_0), b(x(\cdot)) + W(x(\cdot))u(\cdot)).$$

The second mapping, $\mathcal{R}_y : \mathbb{R}^n \times \mathcal{F}_u \to \mathbb{R}^n \times \mathbb{R}^n$, is the linear mapping defined as

$$(y_0, \omega(\cdot)) \mapsto y(T)$$

and given by

$$y(T) = \phi_y(T; y_0, \omega(\cdot)).$$

Here ϕ_y is the state transition function for the system (4.9).

The third mapping is defined as $y(T) \mapsto x(T)$ and is simply the inverse of Ψ.

The second mapping, $\mathcal{R}_y$, is linear, therefore it coincides with its differential. It is clearly surjective due to the (uniform) controllability of the linear time-invariant system (4.9). Using the fact that the system is feedback linearizable on $\overline{\mathcal{XU}}$, we can conclude that the mappings $\mathcal{B}$ and Ψ^{-1}, and their differentials exist and are invertible. The proof follows.

In order to prove uniform linear controllability, we explicitly compute the variation (β, ν) of a trajectory (y, ω) as function of the variation (z, v) of the trajectory (x, u) of the original system. It can be written as

$$\begin{aligned}
\beta &= \Psi(x+z) - \Psi(x) = D\Psi(x)z + o(\|z\|^2) \\
&= T(x)z + o(\|z\|^2) \\
\nu &= b(x+z) + W(x+z)(u+v) - b(x) - W(x)(u) \\
&= \left(Db(x) + \sum_{l=1}^{m} DW_l(x)u_l \right) z + W(x)v + o(\|(z,v)\|^2) \\
&= S(x,u)z + W(x)v + o(\|(z,v)\|^2),
\end{aligned}$$

where (β, ν) satisfies

$$\dot{\beta} = A\beta + B\nu \tag{4.11}$$

and the matrices $T(x) = D\Psi(x) \in \mathbb{R}^{n \times n}$ and $W(x) \in \mathbb{R}^{m \times m}$ are invertible with continuous inverse on $\overline{X}$.

Using the above equations we can write the dynamics of z as

$$\dot{z} = \left(T^{-1}(x)AT(x) + T^{-1}(x)BS(x)\dot{T}^{-1}(x)T(x) \right) z + BW(x)v + o(\|(z,v)\|^2)$$

Clearly the dynamics of the first variation of (x, u), that is the variational equation, is given by

$$\dot{z} = (T^{-1}(x)AT(x) + T^{-1}(x)BS(x) + \dot{T}^{-1}(x)T(x))z + BW(x)v. \tag{4.12}$$

Now we use compactness of $\overline{\mathcal{XU}}$ and Lemma 4.8. First, it is clear that a controllable linear time-invariant system is δ-uniformly controllable for any δ. Then, take a $\delta > 0$ arbitrarily, there exists γ_δ^* such that the following holds for the system in (4.11). For any $t \in [0, T[$ and any $\bar{\beta} \in \mathbb{R}^n$, there exists ω defined on $]t-\delta, t[$ such that, if $\beta(t-\delta) = 0$, then $\beta(t) = \bar{\beta}$ and $\|\omega(\tau)\| \leq \gamma_\delta^* \|\bar{\beta}\|$ for all $\tau \in]t-\delta, t[$.

Now, consider the same δ. We want to find γ_δ satisfying the condition of Lemma 4.8 for (4.12). For any $t \in [0, T[$ and any $\bar{z} \in \mathbb{R}^n$, there exists $v(\cdot)$ and $\bar{\beta} \in \mathbb{R}^n$ such that, if $z(t-\delta) = 0$, then $z(t) = \bar{z} = T^{-1}(x(t-\delta))\bar{\beta}$, and $\|v(\tau)\| \leq \|W^{-1}(x(\tau))\|(\gamma_\delta^*\|T(x(\tau))\| + \|S(x(\tau), u(\tau))\|)\|\bar{z}\|$. Since $T(x)$, $W^{-1}(x)$ and $S(x,u)$ are continuous functions of x and u over the compact set $\overline{\mathcal{XU}}$, there exist $l_T > 0$, $l_W > 0$ and $l_S > 0$ such that $\|v(\tau)\| \leq l_W(\gamma_\delta^* l_T + l_S)\|\bar{z}\|$ for all $\tau \in]t-\delta, t[$. Then we can set $\gamma_\delta = l_W(\gamma_\delta^* l_T + l_S)$ to complete the proof. ■

4.3.2 Operating region and sampled controls

In order to state our next result, we need to introduce some more notation. For control affine systems, let $\mathscr{G}$ be the linear vector space defined as follows

$$\mathscr{G}(x,u) = \operatorname{span}\{g_1(x), \cdots, g_m(x), \operatorname{ad}_{\overline{f}}^k(g_1(\cdot))(x), \cdots, \operatorname{ad}_{\overline{f}}^k(g_m(\cdot))(x), k \geq 1\}, \tag{4.13}$$

where $\overline{f}(x,u) = f_0(x) + g(x)u$.

Then, let $\Phi_{x_0,u_0}(t_0,\tau)$ and $B_{x_0,u_0}(\tau)$ be the state-transition and input matrices of the linear time-varying system obtained by linearizing the nonlinear system (4.5) about the trajectory starting at $x(t_0) = x_0$, $x_0 \in X \subset \mathbb{R}^n$, and generated by the constant control $u(\cdot) = u_0$, $u_0 \in U \subset \mathbb{R}^m$. We define the mapping $(x_0,u_0) \mapsto W_\delta(x_0,u_0)$ as follows

$$W_\delta(x_0,u_0) = \int_{t_0}^{t_0+\delta} \Phi_{x_0,u_0}(t_0,\tau)B_{x_0,u_0}(\tau)B_{x_0,u_0}^T(\tau)\Phi_{x_0,u_0}^T(t_0,\tau)d\tau.$$

The following lemma states that W_δ is a continuous mapping.

Lemma 4.14 (Continuity of the grammian w.r.t. parameters) *Let Σ be a control-affine system as in* (4.5)*. There exists $\delta_1 > 0$ such that for any $0 < \delta < \delta_1$, given any $(x_0,u_0) \in \mathcal{XU}$, the mapping $W_\delta : X \times U \to \mathbb{R}^{n\times n}$, defined as above, is well defined and continuous at (x_0,u_0).*

Proof: From Lipschitz condition on $f_0(x)$ and $g(x)$ we know that there exists δ_1 such that the solution of the nonlinear system exists on $[0,\delta_1[$. This ensures that the integral is well defined. Continuity follows by continuity of the nonlinear and linearized systems with respect to parameters.

■

We are ready to provide sufficient conditions for uniform linear controllability of nonlinear control-affine systems driven by piecewise constant inputs.

Lemma 4.15 (ULC by piecewise constant controls) *Let $\mathcal{XU} = X \times U$, where $X \subset \mathbb{R}^n$ and $U \subset \mathbb{R}^m$ are open, bounded and simply connected sets and $\overline{\mathcal{XU}} = \overline{X} \times \overline{U}$ is the compact closure of $\mathcal{XU}$. Furthermore, let U be convex. Suppose that, given the control affine system in (4.5), rank $\mathscr{G}(x,u) = n$ for any $(x,u) \in \overline{\mathcal{XU}}$. Then*

(i) every bounded trajectory on $[0,\delta]$, $\delta > 0$, arising from $x_0 \in \overline{X}$ and $u(\cdot) \equiv u_0$, $u_0 \in \overline{U}$, is linearly controllable;

(ii) for fixed $\delta_u > 0$, every trajectory on $t \geq 0$, generated by a δ_u-piecewise constant control and remaining in $\mathcal{XU}$ for all $t \geq 0$, is δ-uniformly linearly controllable for any $\delta > 0$.

Proof: The first part of the proof, regarding the linear controllability by constant controls, is standard and can be found for example in [34]. We recall it here for completeness. For general $(x(\cdot), u(\cdot))$, the variational equation is given by

$$\dot{z}(t) = A(x(t), u(t))z(t) + B(x(t))v(t),$$

with the usual expressions $A(x,u) = D_x f_0(x) + \sum_{l=1}^{m} D_x g_l(x) u_l$ and $B(x) = g(x)$. For constant $u(\cdot)$, both $A(x(t), u(t))$ and $B(x(t))$ are smooth ($f_0(x)$ and $g(x)$ are smooth). A sufficient condition for controllability of the linear time-varying system (linear controllability) is given by impulsive linear controllability at $t = 0$. That is

$$\operatorname{rank}\{B(0), (\mathcal{A}^k B(\cdot))(0),\ k \geq 1\} = n, \tag{4.14}$$

where $\mathcal{A} = (\frac{d}{dt} - A(\cdot))$.

This relies on the fact that

$$\frac{d^k}{d\tau^k} \left(\Phi(0,\tau), B_l(\tau)\right) = \left(\mathcal{A}^k B_l(\cdot)\right)(\tau).$$

For the special form of $A(t)$, $B(t)$ as above, we compute

$$(\mathcal{A} B_l(\cdot))(0) = \left[\bar{f}(\cdot, u), g_l(\cdot)\right](x_0). \tag{4.15}$$

For constant $u(\cdot) \equiv u_0$, inductively from (4.15)

$$(\mathcal{A}^k B_l(\cdot))(0) = \operatorname{ad}^k_{\bar{f}(\cdot,u_0)}(g_l(\cdot))(x_0). \tag{4.16}$$

But this is just the condition $\mathscr{G}(x_0, u_0)$ to be full rank for any $(x_0, u_0) \in \overline{\mathcal{XU}}$, thus the proof follows.

As regards statement (ii), we need to prove that fixing $\delta > 0$ arbitrarily, there exists a $k_\delta > 0$ such that, for any $t_0 \geq 0$, then

$$\int_{t_0}^{t_0+\delta} \Phi(t_0,\tau)B(\tau)B^T(\tau)\Phi^T(t_0,\tau)d\tau > k_\delta I.$$

Since $f_0(x)$ and $g(x)$ are locally Lipschitz continuous (in x) it is clear that there is a δ_1 such that for each $(x_0,u_0) \in \overline{\mathcal{XU}}$, the trajectory

$$\phi(t;x_0,u_0)$$

is bounded on $[0,\delta_1]$.

It follows that for each $\delta \in]0,\delta_1[$, there is a k_δ such that

$$W_\delta(x_0,u_0) \geq k_\delta I$$

for all $(x_0,u_0) \in \overline{\mathcal{XU}}$. The desired result follows easily.

■

4.4 Sampled controls and trajectories approximation

In this section we show that sampled-control state trajectories converge uniformly to bounded-control state trajectories.

In order to prove the convergence, we need to introduce a functional norm that we will call *sup average norm*.

Definition 4.16 (Sup average norm) *Let $u(\cdot)$ be a Lebesgue integrable function on $[0,T]$ with values in $\mathbb{R}^m$. We define the sup average norm as the function norm*

$$\|u(\cdot)\|_{sa} = \max_{t\in[0,T]} \left\| \int_0^t u(\tau)\, d\tau \right\|$$

where $\|\cdot\|$ is any norm on $\mathbb{R}^m$. □

Clearly

$$\|u(\cdot)\|_{sa} = \|U(\cdot)\|_\infty$$

for the absolutely continuous function $U(t) = \int_0^t u(\tau)\,d\tau$, where $\|\cdot\|_\infty$ is the sup norm. It is also worth noting that $\|u(\cdot)\|_{sa} \leq \|u(\cdot)\|_1$.

Let $B(\cdot)$ be an absolutely continuous function on $[0,T]$ with values in $\mathbb{R}^{n\times m}$ so that $B(t) = B(0) + \int_0^t \dot{B}(\tau)\,d\tau$ where the derivative $\dot{B}(t)$ is well defined almost everywhere in $[0,T]$. Consider the linear mapping $\mathcal{B} : u(\cdot) \mapsto y(\cdot)$ given by

$$y(t) = \int_0^t B(\tau)\,u(\tau)\,d\tau\,. \tag{4.17}$$

It is easy to see that

$$\|y(\cdot)\|_\infty \leq \|B(\cdot)\|_\infty\,\|u(\cdot)\|_1\,.$$

We may also obtain an estimate depending on $\|u(\cdot)\|_{sa}$. In particular the following lemma holds.

Lemma 4.17 (Sup average norm bound for linear functionals) *Let $B(\cdot)$ be an absolutely continuous function on $[0,T]$ with values in $\mathbb{R}^{n\times m}$ so that $B(t) = B(0) + \int_0^t \dot{B}(\tau)\,d\tau$ where the derivative $\dot{B}(t)$ is well defined almost everywhere in $[0,T]$. Consider the linear mapping $\mathcal{B}$ defined in (4.17). Then*

$$\|y(\cdot)\|_\infty \leq \Big(\|B(\cdot)\|_\infty + \|\dot{B}(\cdot)\|_1\Big)\;\|u(\cdot)\|_{sa}\,.$$

Proof: Set $U(t) = \int_0^t u(\tau)\,d\tau$ and note that

$$y(t) = \int_0^t B(\tau)\,u(\tau)\,d\tau = B(t)\,U(t) - \int_0^t \dot{B}(\tau)\,U(\tau)\,d\tau$$

so that

$$\|y(\cdot)\|_\infty \leq \Big(\|B(\cdot)\|_\infty + \|\dot{B}(\cdot)\|_1\Big)\;\|u(\cdot)\|_{sa}\,.$$

Integration by parts is applicable since $\dot{B}(\cdot)$ and $u(\cdot)$ are (Lebesgue) integrable with absolutely continuous integrals $B(\cdot)$ and $U(\cdot)$. ■

Remark 4.18 (Lebesgue integrability and L_1 norm) *Notice that $u(\cdot)$ is* Lebesgue integrable *iff $u(\cdot) \in L_1$.* □

Now, we show trajectory convergence. Consider a control affine system as in (4.5), where the vector fields $f_0(\cdot)$, $g_i(\cdot)$ are Lipschitz continuous so that

$$\|f_0(x) - f_0(y)\| \leq l_f\|x-y\|,\quad \|g(x) - g(y)\| \leq l_g\|x-y\|$$

for constants l_f, l_g. Fix the initial condition $x(0) = x_0$ and the interval of consideration $[0, T]$ and suppose that the set of allowable controls is bounded with $\|u(t)\| \le b_u$. Then, $f_0(\cdot)$ and $g(\cdot)$ are bounded on the compact reachable region with $\|f_0(x)\| \le b_f$ and $\|g(x)\| \le b_g$. When $f_0(\cdot)$ and $g(\cdot)$ are only locally Lipschitz, the constants l_f, l_g, b_f, b_g will be well defined on the reachable region for sufficiently small T.

Let $u^1(\cdot)$ and $u^2(\cdot)$ be controls bounded by b_u and let $x^1(\cdot)$ and $x^2(\cdot)$ be the corresponding state trajectories starting from x_0. Using $\delta x(\cdot) = x^2(\cdot) - x^1(\cdot)$ and $\delta u(\cdot) = u^2(\cdot) - u^1(\cdot)$, we have

$$\begin{aligned}\|\delta x(t)\| &= \left\| \int_0^t \{(f_0(x^2(\tau)) + g(x^2(\tau))u^2(\tau)) - (f_0(x^1(\tau)) + g(x^1(\tau))u^1(\tau))\}\, d\tau \right\| \\ &\le \int_0^t (l_f + l_g b_u)\|\delta x(\tau)\|\, d\tau + \left\| \int_0^t g(x^2(\tau))\delta u(\tau)\, d\tau \right\| .\end{aligned}$$

Set $B(t) = g(x^2(t))$ and note that

$$\|B(\cdot)\|_\infty \le b_g$$

and

$$\|\dot{B}(\cdot)\|_{1,t} \le l_g(b_f + b_g b_u)t$$

where $\|\cdot\|_{1,t}$ is the truncated $L_1(0,t)$ norm. Note that these estimates do not require $g(\cdot)$ to be differentiable, Lipschitz is sufficient. Continuing, we have

$$\|\delta x(t)\| \le \int_0^t (l_f + l_g b_u)\|\delta x(\tau)\|\, d\tau + (b_g + l_g(b_f + b_g b_u)t)\|\delta u(\cdot)\|_{sa,t}$$

where we have used a truncated norm $\|\cdot\|_{sa,t}$. Using the Bellman-Gronwall lemma, we find that

$$\|\delta x(t)\| \le (b_g + l_g(b_f + b_g b_u)t)\, e^{(l_f + l_g b_u)t}\, \|\delta u(\cdot)\|_{sa,t}$$

where we have used the fact that final term in the previous estimate is strictly increasing and so that, for each t, the maximum value (occuring at t) may be used as a constant in the lemma. Thus

$$\|\delta x(\cdot)\|_\infty \le L\, \|\delta u(\cdot)\|_{sa}$$

with $L = (b_g + l_g(b_f + b_g b_u)T)\, e^{(l_f + l_g b_u)T}$.

We wish to consider the approximation of a bounded function $u(\cdot)$ by control functions $u^k(\cdot)$ that are constant on intervals of length $T/2^k$ for integers

$k \geq 0$. Let $u^k_{avg}(\cdot)$ be the sampled function obtained by averaging $u(\cdot)$ over intervals of length $T/2^k$ so that, for instance, $u^0_{avg}(\cdot)$ is the constant function with value $u_{avg} = (1/T)\int_0^T u(\tau)\, d\tau$. It is easy to show that

$$\|u^k_{avg}(\cdot) - u(\cdot)\|_{sa} \leq Tb_u/2^{k+1}\,.$$

That is, $\|u^\delta_{avg}(\cdot) - u(\cdot)\|_{sa} \leq b_u\delta/2$ for sample period δ, independent of interval length T.

The above arguments prove the following proposition.

Proposition 4.19 *The sampled-control state trajectories converge uniformly to the actual state trajectory with a rate that is at least linear in the sampling period.* □

Remark 4.20 (A conjecture) *We believe that this proposition could be a preliminary step on the way of finding sufficient conditions for uniform linear controllability of control affine systems driven by essentially bounded inputs.* □

4.5 Discussion

We have introduced the novel notion of *operating region* meant as a region where trajectories are ensured to be uniformly linearly controllable and thus exponentially stabilizable. The characterization of this region for control affine systems is still preliminary. For future work, we aim to provide sufficient conditions to characterize an operating region for control affine systems driven by (essentially) bounded inputs.

Chapter 5

Network models

In this chapter we introduce the mathematical model that we will use in the next chapters to deal with networks. After a brief review on graph definitions and main properties, we introduce a model for a network of processors. This model is inspired by the work in [43]. Then, we define a more complex network model, where the nodes are dynamical systems, e.g. mobile robots. We use the formal model introduced in [47] modified for the discrete time case.

5.1 Introduction

Motion coordination is an emerging discipline that combines control and communication problems. For this reason, standard definitions and models from communication theory or from control theory are not suitable to model the new entities involved in this discipline. The scope of this chapter is to provide a good model to work with in studying motion coordination problems.

Notation

We let $\mathbb{N}$, $\mathbb{N}_0$, and $\mathbb{R}_+$ denote the natural numbers, the non-negative integer numbers, and the positive real numbers, respectively. We let $\prod_{i\in\{1,\dots,n\}} S_i$ denote the Cartesian product of sets $S_1, \dots, S_n$. For $p \in \mathbb{R}$, we let $\lfloor p \rfloor$ and $\lceil p \rceil$ denote the floor and ceil of p. For $r \in \mathbb{R}_+$ and $p \in \mathbb{R}^d$, we let $B(p, r)$ denote the closed ball centered at p with radius r, i.e., $B(p, r) = \{q \in \mathbb{R}^d \mid \|p - q\|_2 \leq r\}$ For $f, g : \mathbb{N} \to \mathbb{R}$, we say that $f \in O(g)$ (respectively,

$f \in \Omega(g)$) if there exist $n_0 \in \mathbb{N}$ and $k \in \mathbb{R}_+$ such that $|f(n)| \leq k|g(n)|$ for all $n \geq n_0$ (respectively, $|f(n)| \geq k|g(n)|$ for all $n \geq n_0$). If $f \in O(g)$ and $f \in \Omega(g)$, then we use the notation $f \in \Theta(g)$.

5.2 Preliminaries on graphs

In this section we introduce the notion of directed graph that will be a key concept in the definition of network models.

We let $\mathcal{G} = (V, E)$ denote a directed graph, where $V = \{1, \dots, n\}$ is the set of nodes (or vertices) of the graph and $E : V \to V \times V$ is the edge map describing the set of directed edges $E(V)$ of the graph. For each node i of $\mathcal{G}$, the number of edges going out from (coming into) node i is called *in-degree* (*out-degree*) and is denoted $\texttt{indeg}^{[i]}$ ($\texttt{outdeg}^{[i]}$). The set of outgoing (incoming) neighbors of node i are the set of nodes to (from) which there are edges from (to) i. They are denoted $\mathcal{N}_O(i)$ and $\mathcal{N}_I(i)$, respectively. A direct graph is called *strongly connected* if for every pair of nodes $(i, j) \in V \times V$, there exists a path of directed edges that goes from i to j. The minimum number of edges between node i and j is called the *distance from i to j* and is denoted $\mathrm{dist}(i, j)$. The maximum $\mathrm{dist}(i, j)$ taken over all pairs (i, j) is the *diameter* and is denoted $\mathrm{diam}(\mathcal{G})$. Graphs with undirected edges are *undirected graphs*. Such graphs can be considered a special case of the directed graphs defined above, in the sense that they are equivalent to directed graphs with bidirectional edges between all pairs of neighbors. In this case, for each node i of $\mathcal{G}$, we simply talk of *degree*, $\texttt{deg}^{[i]}$, and set of neighbors, $\mathcal{N}(i)$, of node i.

The definition of graph given above does not take into account the evolution of the graph with respect to time. It may be imagined just as a static picture. In order to take into account the time-evolution of the graph we may allow the edge map to depend on time. In this case the graph is called *time-dependent*. If the graph does not depend on time we will call it *time-independent*.

If the graph is time-dependent different notions of connectivity may arise. We could ask the graph to be strongly connected at each time instant. However, this is a quite strong requirement that could be hard to satisfy in some applications and that is not necessary for many properties. A weaker notion of connectivity for time-dependent graphs is the so called *joint connectivity*. Roughly speaking, a graph is said to be jointly connected if for every instant there exists a time interval in the future such that the graph obtained as union of the graphs in that interval is connected. A stronger notion is the *uniform joint connectivity*. It requires that the length of the interval where the graph is connected is fixed, that is it does not depend on the instant of time. Formally we have the following definition.

Definition 5.1 (Connectivity notions) *Let $t \mapsto \mathcal{G}(t) = (V, E(t))$ be a time-dependent graph. The graph $\mathcal{G}$ is said* jointly strongly connected *if for every $t \in \mathbb{R}$,*

$$\cup_{\tau=t}^{+\infty} \mathcal{G}(\tau)$$

is strongly connected. Moreover, the graph $\mathcal{G}$ is said uniformly strongly connected *if there exists $S > 0$ such that for every $t \in \mathbb{R}$*

$$\mathcal{G}_S = \cup_{\tau=t}^{t+S} \mathcal{G}(\tau)$$

is strongly connected. □

Graphs could depend not only on time, but also on the "state" where the nodes live. We call such graphs *state-dependent*. We introduce this notion only for undirected graphs. For a set X, let $\mathbb{F}(X)$ be the collection of finite subsets of X; e.g., $\mathcal{P} \in \mathbb{F}(\mathbb{R}^d)$ is a set of points. For a finite set X, let $\mathbb{G}(X)$ be the set of undirected graphs whose vertices are elements of X, i.e., whose vertex set belongs to $\mathbb{F}(X)$. For a set X, a *state dependent graph on* X is a map $\mathcal{G} : \mathbb{F}(X) \to \mathbb{G}(X)$ that associates to a finite subset V of X an undirected graph with vertex set V and edge set $\mathcal{E}_{\mathcal{G}}(V)$ where $\mathcal{E}_{\mathcal{G}} : \mathbb{F}(X) \to \mathbb{F}(X \times X)$ satisfies $\mathcal{E}_{\mathcal{G}}(V) \subseteq V \times V$. In other words, what edges exist in $\mathcal{G}(V)$ depends on the elements of V that constitute the nodes.

An example of state dependent graph that will be widely used in the sequel is the *disk graph*. Given $r_{\text{pos}} \in \mathbb{R}_+$, the disk graph $\mathcal{G}_{\text{disk}}(r_{\text{pos}})$ is the state dependent graph on $\mathbb{R}^d$ defined as follows: for $\{p_1, \ldots, p_n\} \subset \mathbb{R}^d$, the pair (p_i, p_j) is an edge in $\mathcal{G}_{\text{disk}}(r_{\text{pos}}) \cdot (\{p_1, \ldots, p_n\})$ if and only if

$$\|p_i - p_j\|_2 \leq r_{\text{pos}} \qquad \Longleftrightarrow \qquad p_i - p_j \in B(0_d, r_{\text{pos}}).$$

Sometimes, with some abuse of notation, we will use (i, j), instead of (p_i, p_j), to indicate the edge between node i and node j.

5.3 Network of processors

Following [43], we define a synchronous network system as a "collection of *computing elements* located at nodes of a directed network graph." These computing elements are sometimes called *processors*, thus suggesting that they are piece of hardware. However it is useful to think of them as *logical processes* running on (but not identical to) the actual hardware processors.

A synchronous network is characterized by two elements: the communication graph and the processes associated with each node of the communication graph. A *communication graph* is a directed graph $\mathcal{G} = (I, E_{\text{cmm}})$, where $I = \{1, \dots, n\}$ is the set of identifiers of the computing elements (processes) and $E_{\text{cmm}} : \mathbb{N} \times I \to I \times I$ is called *communication edge map* and is such that

$$E(t, I) = \{(i, j) \in I \times I \mid \text{process } i \text{ cancommunicate to } j \text{ at time } t\}.$$

If the communication edge map depends explicitly on time, then the network is said to be *time-dependent*, otherwise we call it time-independent. Associated with each directed edge $(i, j) \in \mathcal{G}$ there is a *channel*, also known as *link* which is a location that can, at any time, hold at most a single message of a certain message alphabet M. We underline the fact that in this general model each edge, and therefore the associated channel, is directed. This means that the presence of the edge $(i, j) \in \mathcal{G}$ implies that process i can send a message to process j, but not vice-versa.

Next we define the notion of distributed algorithm.

Definition 5.2 *Let $\mathcal{G} = (I, E)$ be a communication graph. A process $i \in I$ consists of the sets*

- *W, set of "logical" states $w^{[i]}$, $i \in I$;*
- *$W_0 \subset W$, subset of allowable initial values;*
- *M, message alphabet, collection of messages $y_j^{[i]} \in M$, $(i, j) \in I \times I$;*

and the maps

- $\text{msg} : W \times I \to M$, *called* message-generation function*;*
- $\text{stf} : W \times M^n \to W$, *called* state-transition function. □

The above definition may be explained as follows. Each process has a set of logical states, among which it can be distinguished a subset of initial states, and an alphabet of possible messages. The message generation function specifies the message that process i sends to its outgoing neighbors based on its current state. The state-transition function specifies, given the current state and the collection of messages of the incoming neighbors, the new state to which each process moves.

Execution of the network begins with all processes in their start states and all channels empty. Then the processes repeatedly perform the following two actions. First, the ith process sends to each of its outgoing neighbors in the communication graph a message (possibly the `null` message) computed by applying the message-generation function to the current value of $w^{[i]}$. At this point new messages are available in each channel. After a negligible period of time, the ith logical process computes the new value of its logical variables $w^{[i]}$ by applying the state-transition function to the current value of $w^{[i]}$, and to the incoming messages (ready in each channel). The combination of the two actions is called *communication round* or simply round.

In the execution scheme described above, we have not mentioned any restrictions on the amount of computation to execute the state-transition and message-generation functions, implicitly assuming that, at each round, each process has sufficient time to execute all the calculations. In the design of the algorithms we will take into account such bounds in two different ways. The first one is to upper bound the execution-time of the algorithm so that it can be always solved during a communication round. The second way to do that is to slightly change the model of the network described above, by allowing the state-transition function to be executed in multiple rounds. In particular we imagine that each processor may operate in multi-tasking mode, so that it can run the message-generation function while keeping executing the state-transition function if the computation has not ended. If that happens, the message is generated by using the logical state of the previous round (which has not been updated yet).

The last aspect to consider is the *process halting*, that is a situation such that the network (and therefore each process) is in a idle mode. It is useful to distinguish such status because it represents an energy-saving status in which the network may stay “indefinitely”. Also, such status can be used to indicate the achievement of a prescribed task. Formally, we say that a process is in *halting status* if the logical state is a fixed point for the state-transition function (that becomes a self-loop) and no message (or equivalently the `null` message) is generated.

5.4 Robotic network

In the following we define a more complex architecture than the simple synchronous network defined in the previous section. We call it *robotic network*. It is a collection of dynamical systems, with a logical process of a suitable communication digraph associated to each system. In other words we can represent a robotic network as in Figure 5.1 with two layers linked to each other: a *physical layer* represented by the dynamical systems and a *logical layer* represented by the communication graph (i.e., by the logical processes communicating according to the communication edge map).

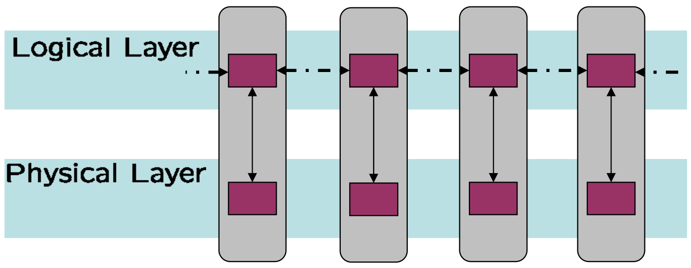


Figure 5.1: Robotic network scheme

We describe a (uniform) network of robotic agents using the formal model introduced in [47] modified for the discrete time case. The network is modeled as a tuple $(I, \mathcal{A}, E_{\text{cmm}})$. $I = \{1, \dots, n\}$ is the *set of unique identifiers (UIDs)*; $\mathcal{A} = \{A^{[i]}\}_{i \in I} = \{(X, U, X_0, f)\}_{i \in I}$ is called the *set of physical agents* and is a set of control systems consisting of a differentiable manifold X (state space), a compact subset U of $\mathbb{R}^m$ (input space), a subset X_0 of X (set of allowable initial states) and a (sufficiently smooth) map $f : X \times U \to X$ describing the dynamics of ith agent; $E_{\text{cmm}} : X^n \to I \times I$ is the *communication edge map*.

The robotic network evolves according to a discrete-time communication and motion model.

Definition 5.3 (Control and communication law) *Let $\mathcal{S}$ be a robotic network. A* (uniform, synchronous, dynamic) control and communication law *$\mathcal{CC}$ for $\mathcal{S}$ consists of the sets M (message alphabet), W (set of logical states) and $W_0 \subseteq W$ (allowable initial values) defined in Definition 5.2 and of the maps:*

(i) $\text{msg} : X \times W \times I \to M$, *called* message-generation function*;*

(ii) $\text{stf} : W \times M^n \to W$, *called* state-transition function*;*

(iii) $\mathrm{ctl} : X \times W \times M^n \to U$, *called* control function. □

In a robotic network, together with the message-generation function and the state-transition function (already defined for networks of processors), a control function is defined. It specifies the new physical state of the (physical) agents based on the current logical and physical state, and on the incoming messages. Notice that, with some abuse of notation, we used the same label for the message-generation function and state-transition function as in the simple network. However, here these two functions may depend also on the physical states $x^{[i]}$. This means that there is a bidirectional relation between the logical layer and the motion layer, as it appears in Figure 5.1.

Roughly speaking, this definition has the following meaning: for all $i \in I$, to the ith physical agent corresponds a logical process, labeled i, that performs the following actions. First, at each communication round the ith logical process sends to each of its outgoing neighbors in the communication graph a message (possibly the `null` message) computed by applying the message-generation function to the current values of $x^{[i]}$ and $w^{[i]}$. After a negligible period of time, the ith process resets the value of its logical state $w^{[i]}$ by applying the state-transition function to the current value of $w^{[i]}$, and to the messages received at time t. Between communication instants, the motion of the ith agent is determined by applying the control function to the current value of $x^{[i]}$, and the current value of $w^{[i]}$. This idea is formalized as follows.

Definition 5.4 (Evolution of a robotic network) *Let $\mathcal{S}$ be a robotic network and $\mathcal{CC}$ be a control and communication law for $\mathcal{S}$. The* evolution *of $(\mathcal{S}, \mathcal{CC})$ from initial conditions $x_0^{[i]} \in X_0$ and $w_0^{[i]} \in W_0$, $i \in I$, is the set of curves $x^{[i]} : \mathbb{N} \to X$ and $w^{[i]} : \mathbb{N} \to W$, $i \in I$, satisfying*

$$x^{[i]}(t+1) = f\big(x^{[i]}(t),\, \mathrm{ctl}(x^{[i]}(t), w^{[i]}(t+1), y^{[i]}(t))\big),$$

where, for $i \in I$,

$$w^{[i]}(t+1) = \mathrm{stf}(w^{[i]}(t), y^{[i]}(t))\,,$$

with the conventions that $x^{[i]}(t_0) = x_0^{[i]}$ and $w^{[i]}(t_0) = w_0^{[i]}$, $i \in I$. Here, the function $y^{[i]} : \mathbb{N} \to M^n$ (describing the messages received by agent i) has components

$$y_j^{[i]}(t) = \begin{cases} \mathrm{msg}(x^{[j]}(t), w^{[j]}(t), i), & \textit{if } (i,j) \in E_{\mathrm{cmm}}, \\ \texttt{null}, & \textit{otherwise.} \end{cases}$$

□

We are ready to define the notion of task and of task achievement by a robotic network.

Definition 5.5 (Coordination task) *Let $\mathcal{S}$ be a robotic network. A* (static) coordination task *for $\mathcal{S}$ is a map $\mathcal{T}: X^n \to \{\texttt{true}, \texttt{false}\}$. Additionally, let $\mathcal{CC}$ a control and communication law for $\mathcal{S}$. The law $\mathcal{CC}$* achieves *the task $\mathcal{T}$ if, for all initial conditions $x_0^{[i]} \in X_0$ and $w_0^{[i]} \in W_0$, $i \in I$, the corresponding network evolution $t \mapsto (x(t), w(t))$ has the property that there exists $T \in \mathbb{N}$ such that $\mathcal{T}(x(t)) = \texttt{true}$ for all $t \geq T$.* □

Chapter 6

Maintaining connectivity in second order wireless networks

In this chapter we consider ad-hoc networks of robotic agents with double integrator dynamics. For such networks, the connectivity maintenance problems are: (i) do there exist control inputs for each agent to maintain network connectivity, and (ii) given desired controls for each agent, can one compute the closest connectivity-maintaining controls in a distributed fashion? The proposed solution is based on three contributions. First, we define and characterize admissible sets for double integrators to remain inside disks. Second, we establish an existence theorem for the connectivity maintenance problem by introducing a novel state-dependent graph, called the *double-integrator disk graph.* Finally, we design a distributed "flow-control" algorithm to compute optimal connectivity-maintaining controls.

6.1 Introduction

The motion coordination problem for groups of autonomous agents is a control problem in the presence of communication constraints. Typically, each agent makes decisions based only on partial information about the state of the entire network that is obtained via communication with its immediate neighbors. One important difficulty is that the topology of the communication network depends on the agents' locations and, therefore, changes with the evolution of the network. In order to ensure a desired emergent behavior for a group of agents, it is necessary that the group does not disintegrate into subgroups that are unable to communicate with each other. In other words, some restrictions must be applied on the movement of the agents to ensure *connectivity* among the members of the group. In terms of design, it is required to constrain the control input such that the resulting topology maintains connectivity throughout its course of evolution. In [4], a connectivity constraint was developed for a group of agents modeled as first-order discrete time dynamic systems. In [4] and in the related references [40, 13], this constraint is used to solve rendezvous problems. Connectivity constraints for line-of-sight communication are proposed in [19]. Another approach to connectivity maintenance for first-order systems is proposed in [58]. In [64], a centralized procedure to find the set of control inputs that maintain k-hop connectivity for a network of agents is given. However, there is no guarantee that the resulting set of feasible control inputs in non-empty. In this chapter we fully characterize the set of admissible control inputs for a group of agents modeled as second order discrete time dynamic systems, which ensures connectivity of the group in the same spirit as described earlier.

The contributions of the chapter are threefold. First, we consider a control system consisting of a double integrator with bounded control inputs. For such a system, we define and characterize the admissible set that allows the double integrator to remain inside disks. Second, we define a novel state-dependent graph – the *double-integrator disk graph* – and give an existence theorem for the connectivity maintenance problem for networks of second order agents with respect to an appropriate version of this new graph. Finally, we consider a relevant optimization problem, where given a set of desired control inputs for all the agents it is required to find the optimal set of connectivity-maintaining control inputs. We cast this problem into a standard quadratic programming problem and provide a distributed "flow-control" algorithm to solve it.

The chapter is organized as follows. In Section 6.2, we define and characterize the admissible sets for a double integrator to remain inside a disk and based on this we define a new graph – the *double-integrator disk graph*. In Section 6.3, we provide an existence theorem for the set of control inputs for the whole network of second order agents that maintains connectivity with respect to an appropriately scaled version of this new graph. We also characterize and give an inner polytopic representation of the constraint set for these connectivity-maintaining control inputs. In Section 6.4, we consider the problem of searching this constraint set for the optimal set of controls in a distributed way. Section 6.5 has some illustrative simulations which also suggest an alternative way of achieving a weak form of flocking of the agents. Finally we conclude with a few remarks about future work in Section 6.6.

6.2 Preliminary developments

We begin with some notations. For $d \in \mathbb{N}$, we let 0_d and 1_d denote the vectors in $\mathbb{R}^d$ whose entries are all 0 and 1, respectively. We let $\|p\|$ denote the Euclidean norm of $p \in \mathbb{R}^d$. For $r \in \mathbb{R}_+$ and $p \in \mathbb{R}^d$, we let $B(p, r)$ denote the closed ball centered at p with radius r, i.e., $B(p, r) = \{q \in \mathbb{R}^d \mid \|p - q\| \le r\}$. For $x, y \in \mathbb{R}^d$, we let $x \preceq y$ denote component-wise inequality, i.e., $x_k \le y_k$ for $k \in \{1, \dots, d\}$. We let $f : A \rightrightarrows B$ denote a set-valued map; in other words, for each $a \in A$, $f(a)$ is a subset of B. We identify $\mathbb{R}^d \times \mathbb{R}^d$ with $\mathbb{R}^{2d}$.

6.2.1 Maintaining a double integrator inside a disk

For $t \in \mathbb{N}_0$, consider the discrete-time control system in $\mathbb{R}^{2d}$

$$\begin{aligned} p(t+1) &= p(t) + v(t), \\ v(t+1) &= v(t) + u(t), \end{aligned} \tag{6.1}$$

where the norm of the control is upper-bounded by $r_{\text{ctr}} \in \mathbb{R}_+$, i.e., $u(t) \in B(0_d, r_{\text{ctr}})$ for $t \in \mathbb{N}_0$. We refer to this control system as the *discrete-time double integrator* in $\mathbb{R}^d$ or, more loosely, as a second-order system. Given $(p, v) \in \mathbb{R}^{2d}$ and $\{u_\tau\}_{\tau \in \mathbb{N}_0} \subseteq B(0_d, r_{\text{ctr}})$, let $\phi(t, (p, v), \{u_\tau\})$ denote the solution of (6.1) at time $t \in \mathbb{N}_0$ from initial condition (p, v) with inputs $u_1, \dots, u_{t-1}$.

In what follows we consider the following problem: assume that the initial position of (6.1) is inside a disk centered at 0_d, find inputs that keep it inside that disk. This task is impossible for general values of the initial velocity. In what follows we identify assumptions on the initial velocity that render the task possible.

For $r_{\text{pos}} \in \mathbb{R}_+$, we define the *admissible set at time zero* by

$$\mathcal{A}_0^d(r_{\text{pos}}) = B(0_d, r_{\text{pos}}) \times \mathbb{R}^d.$$

For $r_{\text{pos}}, r_{\text{ctr}} \in \mathbb{R}_+$, we define the *admissible set for m time steps* by

$$\begin{aligned} \mathcal{A}_m^d(r_{\text{pos}}, r_{\text{ctr}}) = \big\{(p, v) \in \mathbb{R}^{2d} \mid \exists \{u_\tau\}_{\tau \in [0, m-1]} \subseteq B(0_d, r_{\text{ctr}}) \\ \text{s.t. } \phi(t, (p, v), \{u_\tau\}) \in \mathcal{A}_0^d(r_{\text{pos}}) \ \forall t \in [0, m]\big\}, \end{aligned}$$

and the *admissible set* by

$$\begin{aligned} \mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}}) = \big\{(p, v) \in \mathbb{R}^{2d} \mid \exists \{u_\tau\}_{\tau \in \mathbb{N}_0} \subseteq B(0_d, r_{\text{ctr}}) \\ \text{s.t. } \phi(t, (p, v), \{u_\tau\}) \in \mathcal{A}_0^d(r_{\text{pos}}), \ \forall t \in \mathbb{N}_0\big\}. \end{aligned}$$

With slight abuse of notation we shall sometimes suppress the arguments in the definitions of admissible sets. The following theorem establishes some important properties of the admissible sets.

Theorem 6.1 (Properties of the admissible sets) *For all $d \in \mathbb{N}$ and $r_{\mathrm{pos}}, r_{\mathrm{ctr}} \in \mathbb{R}_+$, the following statements hold:*

(i) for all $m \in \mathbb{N}$, $\mathcal{A}^d_m(r_{\mathrm{pos}}, r_{\mathrm{ctr}}) \subseteq \mathcal{A}^d_{m-1}(r_{\mathrm{pos}}, r_{\mathrm{ctr}})$ and

$$\mathcal{A}^d(r_{\mathrm{pos}}, r_{\mathrm{ctr}}) = \lim_{m\to+\infty} \mathcal{A}^d_m(r_{\mathrm{pos}}, r_{\mathrm{ctr}}) = \lim_{m\to+\infty} \cap_{k=1}^m \mathcal{A}^d_k(r_{\mathrm{pos}}, r_{\mathrm{ctr}})\,;$$

(ii) $\mathcal{A}^d(r_{\mathrm{pos}}, r_{\mathrm{ctr}})$ is a convex, compact set and is the largest controlled-invariant[1] subset of $\mathcal{A}^d_0(r_{\mathrm{pos}})$;

(iii) $\mathcal{A}^d(r_{\mathrm{pos}}, r_{\mathrm{ctr}})$ is invariant under orthogonal transformations in the sense that, if $(p, v) \in \mathcal{A}^d(r_{\mathrm{pos}}, r_{\mathrm{ctr}})$, then also $(Rp, Rv) \in \mathcal{A}^d(r_{\mathrm{pos}}, r_{\mathrm{ctr}})$ for all orthogonal[2] matrices R in $\mathbb{R}^{d\times d}$;

(iv) if $0 < r_1 < r_2$, then $\mathcal{A}^d(r_{\mathrm{pos}}, r_1) \subset \mathcal{A}^d(r_{\mathrm{pos}}, r_2)$ and $\mathcal{A}^d(r_1, r_{\mathrm{ctr}}) \subset \mathcal{A}^d(r_2, r_{\mathrm{ctr}})$.

Proof: The two facts in statement (i) are direct consequences of the definitions of $\mathcal{A}^d_m$ and $\mathcal{A}^d$. Regarding statement (ii), each $\mathcal{A}^d_m$, $m \in \mathbb{N}$, is closed, the intersection of closed sets is closed, and, therefore, $\mathcal{A}^d = \lim_{m\to+\infty} \cap_{k=1}^m \mathcal{A}^d_k$ is closed. To show that $\mathcal{A}^d$ is bounded it suffices to show that $\mathcal{A}^d_1$ is bounded. Note that $(p, v) \in \mathcal{A}^d_1$ implies that there exists $u \in B(0_d, r_{\mathrm{ctr}})$ such that $(p, v) \in \mathcal{A}^d_0$ and $(p+v, v+u) \in \mathcal{A}^d_0$. This, in turn, implies that $p \in B(0_d, r_{\mathrm{pos}})$ and $p + v \in B(0_d, r_{\mathrm{pos}})$. Therefore, $\mathcal{A}^d_1$ is bounded. Next, we prove that $\mathcal{A}^d_m$ is convex. Given (p_1, v_1) and (p_2, v_2) in $\mathcal{A}^d_m$, let u_1 and u_2 be controls that ensure that $\phi(t, (p_i, v_i), \{u_i\}) \in \mathcal{A}^d_0$, $t \in [0, m]$, $i \in \{1, 2\}$. For $\lambda \in [0, 1]$, consider the initial condition $(p_\lambda, v_\lambda) = (\lambda p_1 + (1-\lambda)p_2, \lambda v_1 + (1-\lambda)v_2)$ and the input $u_\lambda = \lambda u_1 + (1-\lambda)u_2$, and note that, by linearity,

$$\phi(t, (p_\lambda, v_\lambda), u_\lambda) = \lambda\phi(t, (p_1, v_1), \{u_1\}) + (1-\lambda)\phi(t, (p_2, v_2), \{u_2\}), \quad t \in [0, m].$$

Because $\phi(t, (p_1, v_1), \{u_1\})$ and $\phi(t, (p_2, v_2), \{u_2\})$ belong to the convex set $\mathcal{A}^d_0$, then also their convex combination does. Therefore, (p_λ, v_λ) belongs to

[1]A set is controlled invariant for a control system if there exists a feedback law such that the set is positively invariant for the closed-loop system.

[2]A matrix $R \in \mathbb{R}^{d\times d}$ is orthogonal if $RR^T = R^T R = I_d$.

$\mathcal{A}^d_m$, and $\mathcal{A}^d_m$ is convex. Finally, $\mathcal{A}^d$ is convex because the intersection of convex sets is convex.

Next, we show that $\mathcal{A}^d$ is controlled invariant. Given $(p, v) \in \mathcal{A}^d$ (with corresponding control sequence $\{u_\tau\}_{\tau\in\mathbb{N}_0}$), we need to show that there exists a control input $x \in B(0_d, r_{\rm ctr})$ such that $\phi(1,(p,v),x) \in \mathcal{A}^d$. Such input can be chosen as $x = u_0$. Indeed, the control sequence $\{u_{\tau+1}\}_{\tau\in\mathbb{N}_0}$ keeps the trajectory starting from $\phi(1,(p,v),x)$ inside $\mathcal{A}^d_0$ and, therefore, $\phi(1,(p,v),x) \in \mathcal{A}^d$. Additionally, it is easy to see that $\mathcal{A}^d \subset \mathcal{A}^d_0$. Finally, we need to prove that $\mathcal{A}^d$ is the largest controlled-invariant subset of $\mathcal{A}^d_0$. Assume that there exists $\mathcal{A}^{d*}$ with the same properties and larger than $\mathcal{A}^d$. This means that there exists $(p, v) \in \mathcal{A}^{d*} \setminus \mathcal{A}^d$. This is equivalent to saying that $\exists\ \tau^* \in \mathbb{N}_0$ such that, for every choice of the input u, $\phi(\tau^*,(p,v),u) \notin \mathcal{A}^d_0$. But, since $\mathcal{A}^{d*} \subset \mathcal{A}^d_0$, this leads to a contradiction.

Regarding statement (iii), observe that, if $(p, v) \in \mathcal{A}^d_0$, then $(Rp, Rv) \in \mathcal{A}^d_0$ and, if $u \in B(0, r_{\rm ctr})$, then $Ru \in B(0, r_{\rm ctr})$. Therefore, using again the linearity of the maps ϕ, the proof follows. Regarding statement (iv), the two results follow from the definition of $\mathcal{A}^d(r_{\rm pos}, r_{\rm ctr})$ and the facts that, for all $0 < r_1 < r_2$, $B(0, r_1) \subset B(0, r_2)$ and $\mathcal{A}^d_0(r_1) \subset \mathcal{A}^d_0(r_2)$. ■

Next, we study the set-valued map that associates to each state in $\mathcal{A}^d(r_{\rm pos}, r_{\rm ctr})$ the set of control inputs that keep the state inside $\mathcal{A}^d(r_{\rm pos}, r_{\rm ctr})$ in one step. We define the *admissible control set* $\mathcal{U}^d(r_{\rm pos}, r_{\rm ctr}) : \mathcal{A}^d(r_{\rm pos}, r_{\rm ctr}) \rightrightarrows B(0_d, r_{\rm ctr})$ by

$$\mathcal{U}^d(r_{\rm pos}, r_{\rm ctr}) \cdot (p, v) = \{u \in B(0_d, r_{\rm ctr}) \mid (p + v, v + u) \in \mathcal{A}^d(r_{\rm pos}, r_{\rm ctr})\},$$

or, equivalently,

$$\mathcal{U}^d(r_{\rm pos}, r_{\rm ctr}) \cdot (p, v) = B(0_d, r_{\rm ctr}) \cap \{w - v \mid (p + v, w) \in \mathcal{A}^d(r_{\rm pos}, r_{\rm ctr})\}. \quad (6.2)$$

Lemma 6.2 (Properties of the admissible control set) *For all $(p, v) \in \mathcal{A}^d(r_{\rm pos}, r_{\rm ctr})$, the set $\mathcal{U}^d(r_{\rm pos}, r_{\rm ctr}) \cdot (p, v)$ is non-empty, convex and compact. For generic $(p, v) \in \mathcal{A}^d(r_{\rm pos}, r_{\rm ctr})$, the set $\mathcal{U}^d(r_{\rm pos}, r_{\rm ctr}) \cdot (p, v)$ does not contain 0_d.*

Proof: The non-emptiness of the set $\mathcal{U}^d(r_{\rm pos}, r_{\rm ctr}) \cdot (p, v)$ derives directly from the definition of $\mathcal{A}^d(r_{\rm pos}, r_{\rm ctr})$. Clearly, from equation (6.2), $\mathcal{U}^d(r_{\rm pos}, r_{\rm ctr}) \cdot (p, v)$ is closed (it is the intersection of two closed sets). It is also bounded ($\mathcal{U}^d(r_{\rm pos}, r_{\rm ctr}) \cdot (p, v) \subset B(0_d, r_{\rm ctr})$), hence it is compact. To prove that it is convex, we need to show the following: given $(p, v) \in \mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$, if there exist u_1 and u_2 in $\mathcal{U}^d(r_{\text{pos}}, r_{\text{ctr}}) \cdot (p, v)$ such that $\phi(1, (p, v), u_1)$ and $\phi(1, (p, v), u_2)$ belong to $\mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$, then $u_{12} = \lambda u_1 + (1 - \lambda)u_2$, $\lambda \in [0, 1]$, belongs to $\mathcal{U}^d(r_{\text{pos}}, r_{\text{ctr}}) \cdot (p, v)$, that is, $\phi(1, (p, v), u_{12}) \in \mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$. But this fact follows directly from the linearity of ϕ and the convexity of $\mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$. This proves that $\mathcal{U}^d(r_{\text{pos}}, r_{\text{ctr}}) \cdot (p, v)$ is convex. The fact that it does not necessarily contain the origin can be proven by contradiction as follows. Consider a $(p, v) \in \mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$ such that $v \neq 0_d$ and $\mathcal{U}^d(r_{\text{pos}}, r_{\text{ctr}}) \cdot (p, v)$ contains 0_d. This means that $(p + v, v)$ also belongs to $\mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$. Now, either $\mathcal{U}^d(r_{\text{pos}}, r_{\text{ctr}}) \cdot (p + v, v)$ does not contain 0_d, in which case we have proved the statement, or $\mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$ also contains $(p + 2v, v)$. Continuing along these lines, if it were true that $\mathcal{U}^d(r_{\text{pos}}, r_{\text{ctr}}) \cdot (p, v)$ contains the origin for all $(p, v) \in \mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$, then one could show that $(p + tv, v)$ belongs to $\mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$ for all $t \in \mathbb{N}$. However, $\mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$ is bounded by Theorem 6.1. Hence, one can always find a $t^* \in \mathbb{N}$ such that $(p + t^*v, v) \in \mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$ but $(p + (t^* + 1)v, v) \notin \mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$, thereby proving that $\mathcal{U}^d(r_{\text{pos}}, r_{\text{ctr}}) \cdot (p + t^*v, v)$ does not contain 0_d. ∎

6.2.2 Computing admissible sets

We characterize $\mathcal{A}^d$ for $d = 1$ in the following result and we illustrate the outcome in Figure 6.1.

Lemma 6.3 (Admissible set in 1 dimension) *For $r_{\text{pos}}, r_{\text{ctr}} \in \mathbb{R}_+$, the following holds:*

(i) $\mathcal{A}^1(r_{\text{pos}}, r_{\text{ctr}})$ is the polytope containing the points $(p, v) \in \mathbb{R}^2$ satisfying

$$-\frac{r_{\text{pos}}}{m} - \frac{m-1}{2} r_{\text{ctr}} \leq v + \frac{p}{m} \leq \frac{r_{\text{pos}}}{m} + \frac{m-1}{2} r_{\text{ctr}}, \tag{6.3}$$

for all $m \in \mathbb{N}$, and $p \in [-r_{\text{pos}}, r_{\text{pos}}]$;

(ii) If $\widehat{m}(r_{\text{pos}}, r_{\text{ctr}}) \in \mathbb{N}$ is defined by

$$\widehat{m}(r_{\text{pos}}, r_{\text{ctr}}) = \left\lceil -\frac{1}{2} + \sqrt{\frac{1}{4} + \frac{4 r_{\text{pos}}}{r_{\text{ctr}}}} \right\rceil, \tag{6.4}$$

then $\mathcal{A}^1 = \mathcal{A}^1_m = \mathcal{A}^1_{\widehat{m}(r_{\text{pos}}, r_{\text{ctr}})}$, for $m \geq \widehat{m}(r_{\text{pos}}, r_{\text{ctr}})$.

Proof: Regarding statement (i), it suffices to show that, for $m \in \mathbb{N}$, $\mathcal{A}^1_m(r_{\text{pos}}, r_{\text{ctr}})$ is the set of points in $\mathcal{A}^1_{m-1}(r_{\text{pos}}, r_{\text{ctr}})$ that satisfy equation (6.3). If we show that this property holds for all m, then we can use statement (i) of Theorem 6.1 to complete the proof. Consider the set of equations (6.1) for m consecutive time indices after t. The solution of the linear system can be written in terms of the state at instant t as

$$\begin{bmatrix} p(t+m) \\ v(t+m) \end{bmatrix} = \begin{bmatrix} 1 & m \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p(t) \\ v(t) \end{bmatrix} + \sum_{\tau=0}^{m-1} \begin{bmatrix} 1 & (m-1-\tau) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t+\tau), \tag{6.5}$$

where we used the equality

$$A^\tau = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}^\tau = \begin{bmatrix} 1 & \tau \\ 0 & 1 \end{bmatrix}, \quad \tau \in \mathbb{N}.$$

It is clear that the points on the boundary of $\mathcal{A}^1_m$ have the property that the maximum control effort is needed to enforce the constraint. In other words we look for the points $(p(t), v(t)) \in \mathcal{A}^1_0$ with $v(t) \geq 0$ (the case $v(t) \leq 0$ can be solved in a similar way) such that the points $p(t+m) \leq r_{\text{cmm}}$ are reached by using the maximum control effort $u(t+\tau) = -r_{\text{ctr}}$, $\tau \in \{0, \dots, m-1\}$.

Substituting the expression of the control in (6.5) we obtain

$$\begin{aligned} p(t+m) &= p(t) + mv(t) - r_{\text{ctr}} \sum_{\tau=0}^{m-1} (m-1-\tau), \\ v(t+m) &= v(t) - mr_{\text{ctr}}, \end{aligned}$$

and using the equality $\sum_{\tau=0}^{m-1}(m-1-\tau) = \frac{m(m-1)}{2}$, we have

$$\begin{aligned} p(t+m) &= p(t) + mv(t) - r_{\text{ctr}} \frac{m(m-1)}{2}, \\ v(t+m) &= v(t) - mr_{\text{ctr}}, \end{aligned} \tag{6.6}$$

In order to belong to $\mathcal{A}^1_m$, the point $(p(t), v(t))$ must satisfy the constraint $p(t+\tau) \leq r_{\text{cmm}}$, $\tau \in \{1, \dots, m\}$, or equivalently

$$v(t) \leq -\frac{p(t)}{\tau} + \frac{r_{\text{cmm}}}{\tau} + r_{\text{ctr}} \frac{(\tau-1)}{2}, \quad \tau \in \{1, \dots, m\}.$$

Using the same procedure for the points in the half plane $v(t) \leq 0$ (in this case the control is $u(t+\tau) = r_{\text{ctr}}$, $\tau \in \{0, \dots, m-1\}$), it turns out that $\mathcal{A}^1_m$ is equal to the set of all pairs $(p, v) \in \mathcal{A}^1_0$ satisfying

$$-\frac{p}{\tau} - \frac{r_{\text{cmm}}}{\tau} - \frac{\tau-1}{2} r_{\text{ctr}} \leq v \leq -\frac{p}{\tau} + \frac{r_{\text{cmm}}}{\tau} + \frac{\tau-1}{2} r_{\text{ctr}}, \quad \tau \in \{1, \dots, m\}.$$

By using statement (i) of Theorem 6.1 the proof is complete.

Regarding statement (ii), let us consider the case $v(t) \geq 0$ and evaluate the points on the boundary such that $(p(t+m), v(t+m)) = (r_{\text{cmm}}, 0)$, $m \in \mathbb{N}$. These points are obtained by substituting the above value of $(p(t+m), v(t+m))$ in (6.6). The points obtained are (p, v) such that

$$p = r_{\text{cmm}} - m\frac{(m+1)}{2}r_{\text{ctr}}, \qquad m \in \mathbb{N}_0.$$

It is easy to see that $\widehat{m}(r_{\text{pos}}, r_{\text{ctr}})$, as defined in equation (6.4), is the lowest m such that $p \leq -r_{\text{cmm}}$. ■

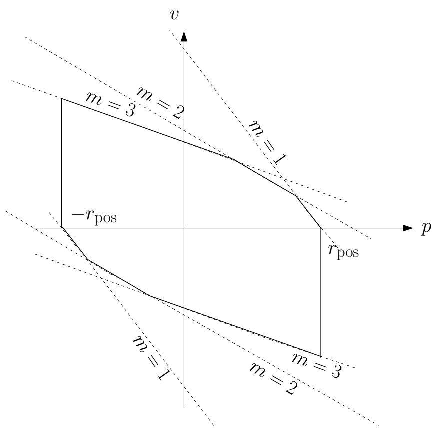


Figure 6.1: The admissible set $\mathcal{A}^1$ for generic values of r_{pos} and r_{ctr}

Remarks 6.4 *(i) If $r_{\text{ctr}} \geq 2r_{\text{pos}}$, then $\mathcal{A}^1 = \mathcal{A}^1_1$, that is, for sufficiently large $r_{\text{ctr}}/r_{\text{pos}}$, the admissible set is equal to the admissible set in* 1 *time step.*

(ii) The methodology for constructing $\mathcal{A}^1(r_{\text{pos}}, r_{\text{ctr}})$ is related to the procedure for constructing the so-called isochronic regions for discrete time optimal control of double integrators, as outlined in [20]. □

Next, we introduce some definitions useful to provide an inner approximation of $\mathcal{A}^d$ when $d \geq 2$. Given $p \in \mathbb{R}^d$ and $v \in \mathbb{R}^d \setminus \{0_d\}$, define $p_{\parallel} \in \mathbb{R}$ and $p_{\perp} \in \mathbb{R}^d$ by

$$p = p_{\parallel}\frac{v}{\|v\|} + p_{\perp},$$

where $p_\perp \cdot v = 0$. For $r_{\text{pos}}, r_{\text{ctr}} \in \mathbb{R}_+$, define

$$\mathcal{A}^d_{\|}(r_{\text{pos}}, r_{\text{ctr}}) = \Big\{(p,v) \in B(0_d, r_{\text{pos}}) \times \mathbb{R}^d \mid v = 0_d \text{ or } (p_\|, \|v\|) \in \mathcal{A}^1\big(\sqrt{r_{\text{pos}}^2 - \|p_\perp\|^2}, r_{\text{ctr}}\big)\Big\}. \quad (6.7)$$

Lemma 6.5 *For $r_{\text{pos}}, r_{\text{ctr}} \in \mathbb{R}_+$, $\mathcal{A}^d_\|(r_{\text{pos}}, r_{\text{ctr}})$ is a compact subset of $\mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$.*

Proof: We begin by showing that definition (6.7) is equivalent to

$$\mathcal{A}^d_\|(r_{\text{pos}}, r_{\text{ctr}}) = \Big\{(p,v) \in \mathcal{A}^d_0 \mid v = 0_d \text{ or } \exists\{u_{\|\tau}\}_{\tau\in\mathbb{N}_0} \subseteq [-r_{\text{ctr}}, r_{\text{ctr}}] \text{ s.t. } \phi\Big(t, (p,v), \{u_{\|\tau}\}\frac{v}{\|v\|}\Big) \in \mathcal{A}^d_0(r_{\text{pos}}), \ \forall t \in \mathbb{N}_0\Big\}. \quad (6.8)$$

To establish this equivalence, we use the definition of the set $\mathcal{A}^1$. For $v \neq 0_d$, we rewrite the solution of the system as

$$\phi(t, (p,v), \{u_\tau\}) = \phi_\|(t, (p,v), \{u_\tau\})\frac{v}{\|v\|} + \phi_\perp(t, (p,v), \{u_\tau\}),$$

where $\phi_\perp(t,(p,v),\{u_\tau\}) \cdot v = 0$ for all $t \in \mathbb{N}_0$. It is easy to see that, if $\{u_\tau\}_{\tau\in\mathbb{N}_0} = \{u_{\|\tau}\}_{\tau\in\mathbb{N}_0}\frac{v}{\|v\|}$, then $\phi_\perp(t,(p,v),\{u_\tau\}) = (p_\perp, 0_d)$ for all $t \in \mathbb{N}_0$. Therefore,

$$\phi(t,(p,v),\{u_\tau\}) = \phi_\|(t,(p,v),\{u_\tau\})\frac{v}{\|v\|} + (p_\perp, 0_d).$$

Note that, if $p = p_\| \frac{v}{\|v\|} + p_\perp$, then $\|p\| \le r_{\text{pos}}$ if and only if $p_\| \le \sqrt{r_{\text{pos}}^2 - \|p_\perp\|^2}$. Therefore, the property $\phi\Big(t,(p,v),\{u_{\|\tau}\}\frac{v}{\|v\|}\Big) \in \mathcal{A}^d_0(r_{\text{pos}})$ is equivalent to

$$\phi_\|\Big(t,(p,v),\{u_{\|\tau}\}\frac{v}{\|v\|}\Big) \in \mathcal{A}^1_0\Big(\sqrt{r_{\text{pos}}^2 - \|p_\perp\|^2}\Big),$$

and, in turn, definitions (6.7) and (6.8) are equivalent. In order to prove that $\mathcal{A}^d_\|(r_{\text{pos}}, r_{\text{ctr}})$ is compact, we simply observe that it is a closed subset of the compact set $\mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}})$. ■

Remark 6.6 *In what follows we use our representation of $\mathcal{A}^d_\|$ to compute an inner approximation for the convex set $\mathcal{A}^d$, for $d \ge 2$. For example, for fixed $p \in B(0_d, r_{\text{pos}})$, we compute velocity vectors v such that $(p,v) \in \mathcal{A}^d$ by considering a sample of unit-length vectors $w \in \mathbb{R}^d$ and computing the maximum allowable velocity v parallel to w according to equation (6.7). Furthermore, we perform computations by adopting inner polytopic representations for the various compact convex sets.* □

6.2.3 The double-integrator disk graph

We are now ready to introduce the notion of double integrator disk graph.

The following three examples of state dependent graphs play an important role. First, given $r_{\text{pos}} \in \mathbb{R}_+$, the *disk graph* $\mathcal{G}_{\text{disk}}(r_{\text{pos}})$ is the state dependent graph on $\mathbb{R}^d$ defined as follows: for $\{p_1, \dots, p_n\} \subset \mathbb{R}^d$, the pair (p_i, p_j) is an edge in $\mathcal{G}_{\text{disk}}(r_{\text{pos}}) \cdot (\{p_1, \dots, p_n\})$ if and only if

$$\|p_i - p_j\| \leq r_{\text{pos}} \quad \iff \quad p_i - p_j \in B(0_d, r_{\text{pos}}).$$

Second, given $r_{\text{pos}}, r_{\text{ctr}} \in \mathbb{R}_+$, the *double-integrator disk graph* $\mathcal{G}_{\text{di-disk}}(r_{\text{pos}}, r_{\text{ctr}})$ is the state dependent graph on $\mathbb{R}^{2d}$ defined as follows: for $\{(p_1, v_1), \dots, (p_n, v_n)\} \subset \mathbb{R}^{2d}$, the pair $((p_i, v_i), (p_j, v_j))$ is an edge if and only if the relative positions and velocities satisfy

$$(p_i - p_j, v_i - v_j) \in \mathcal{A}^d(r_{\text{pos}}, r_{\text{ctr}}).$$

Third, it is convenient to define the disk graph also as a state dependent graph on $\mathbb{R}^{2d}$ by stating that $((p_i, v_i), (p_j, v_j))$ is an edge if and only if (p_i, p_j) is an edge of the disk graph on $\mathbb{R}^d$. We illustrate the first two graphs in Figure 6.2.

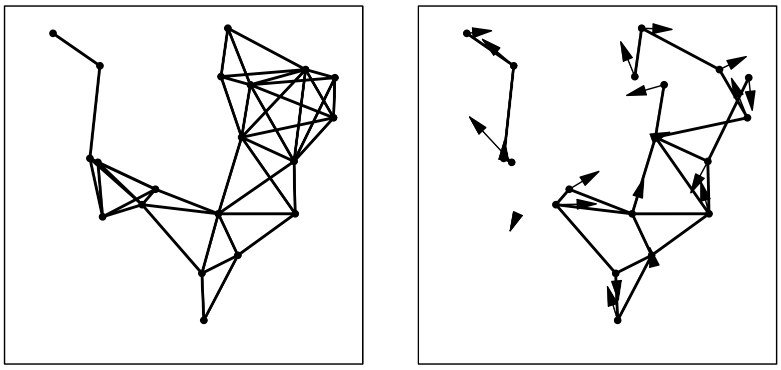


Figure 6.2: The disk graph and the double-integrator disk graph in $\mathbb{R}^2$ for 20 agents with random positions and velocities.

Remark 6.7 *As it is well known, the disk graph is invariant under rigid transformations and reflections. Loosely speaking, the double integrator disk graph is invariant under the following class of transformations: position and velocities of the agents may be expressed with respect to any rotated and translated frame that is moving at constant linear velocity. These transformations are related to the classic Galilean transformations in geometric mechanics.*□

6.3 Connectivity constraints among second-order agents

In this section we state the model, the notion of connectivity, and a sufficient condition that guarantees connectivity can be preserved.

6.3.1 The connectivity maintenance problem

We begin by introducing the notion of *network of robotic agents with second-order dynamics* in $\mathbb{R}^d$. Let n be the number of agents. Each agent has the following computation, motion control, and communication capabilities. For $i \in \{1, \ldots, n\}$, the ith agent occupies a location $p^{[i]} \in \mathbb{R}^d$, moves with velocity $v^{[i]} \in \mathbb{R}^d$, according to the discrete-time double integrator dynamics in (6.1), i.e.,

$$\begin{aligned} p^{[i]}(t+1) &= p^{[i]}(t) + v^{[i]}(t), \\ v^{[i]}(t+1) &= v^{[i]}(t) + u^{[i]}(t), \end{aligned} \tag{6.9}$$

where the norm of all controls $u^{[i]}(t)$, $i \in \{1, \ldots, n\}$, $t \in \mathbb{N}_0$, is upper-bounded by $r_{\text{ctr}} \in \mathbb{R}_+$. The communication model is the following. The processor of each agent has access to the agent location and velocity. Each agent can transmit information to other agents within a distance $r_{\text{cmm}} \in \mathbb{R}_+$. We remark that the control bound r_{ctr} and the communication radius r_{cmm} are the same for all agents.

Remarks 6.8 *(i) Our network model assumes synchronous execution, although it would be important to consider more general asynchronous networks.*

(ii) We will not address the correctness of our algorithms in the presence of measurement errors or communication quantization. □

We now state the control design problem of interest.

Problem 6.9 (Connectivity maintenance) *Choose a state dependent graph $\mathcal{G}_{\text{target}}$ on $\mathbb{R}^{2d}$ and design (state dependent) control constraints sets with the following property: if each agent's control takes values in the control constraint set, then the agents move in such a way that the number of connected components of $\mathcal{G}_{\text{target}}$ (evaluated at the agents' states) does not increase with time.* □

This objective is to be achieved with the limited information available through message exchanges between agents. This problem was originally stated and solved for first-order agents in [4].

6.3.2 A known result for first-order agents

In [4], a connectivity constraint was developed for a set of agents modeled by first-order discrete-time dynamics:

$$p^{[i]}(t+1) = p^{[i]}(t) + u^{[i]}(t).$$

Here the graph whose connectivity is of interest, is the disk graph $\mathcal{G}_{\text{disk}}(r_{\text{cmm}})$ over the vertices $\{p^{[1]}(t), \dots, p^{[n]}(t)\}$. Network connectivity is maintained by restricting the allowable motion of each agent. In particular, it suffices to restrict the motion of each agent as follows. If agents i and j are neighbors in the r_{cmm}-disk graph $\mathcal{G}_{\text{disk}}(r_{\text{cmm}})$ at time t, then their positions at time $t+1$ are required to belong to $B\big(\frac{p^{[i]}(t)+p^{[j]}(t)}{2}, \frac{r_{\text{cmm}}}{2}\big)$. In other words, connectivity between i and j is maintained if

$$\begin{aligned} u^{[i]}(t) &\in B\Big(\frac{p^{[j]}(t) - p^{[i]}(t)}{2}, \frac{r_{\text{cmm}}}{2}\Big), \\ u^{[j]}(t) &\in B\Big(\frac{p^{[i]}(t) - p^{[j]}(t)}{2}, \frac{r_{\text{cmm}}}{2}\Big). \end{aligned}$$

The constraint is illustrated in Figure 6.3.

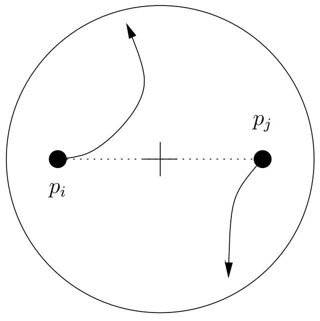


Figure 6.3: Starting from $p^{[i]}$ and $p^{[j]}$, the agents are restricted to move inside the disk centered at $\frac{p^{[i]}+p^{[j]}}{2}$ with radius $\frac{r_{\text{cmm}}}{2}$.

Note that this constraint takes into account only the positions of the agents; this fact can be attributed to the *first-order dynamics* of the agents. We wish to construct a similar constraint for agents with second order dynamics. It is reasonable to expect that this new constraint will depend on positions as well as velocities of the neighboring agents.

6.3.3 An appropriate graph for the connectivity maintenance problem for second-order agents

We begin working on the stated problem with a negative result regarding two candidate target graphs.

Lemma 6.10 (Unsuitable graphs) *Consider a network of n agents with double integrator dynamics* (6.9) *in $\mathbb{R}^d$. Let r_{cmm} be the communication range and let r_{ctr} be the control bound. Let $\mathcal{G}_{\text{target}}$ be either $\mathcal{G}_{\text{disk}}(r_{\text{cmm}})$ on $\mathbb{R}^{2d}$ or $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, 2r_{\text{ctr}})$. There exist states $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$ such that*

(i) the graph $\mathcal{G}_{\text{target}}$ at $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$ is connected, and

(ii) for all $\{u^{[i]}\}_{i\in\{1,\dots,n\}} \subseteq B(0_d, r_{\text{ctr}})$, the graph $\mathcal{G}_{\text{target}}$ at $\{(p^{[i]} + v^{[i]}, v^{[i]} + u^{[i]})\}_{i\in\{1,\dots,n\}}$, is disconnected.

Proof: The proof of the statement for $\mathcal{G}_{\text{target}} = \mathcal{G}_{\text{disk}}(r_{\text{cmm}})$ is straightforward. Consider two agents whose relative position and velocity belong to $\mathcal{A}_0^d \setminus \mathcal{A}_1^d$. Then, after one time step, the two agents will not be connected in $\mathcal{G}_{\text{disk}}(r_{\text{cmm}})$ no matter what their controls are. Next, consider the case $\mathcal{G}_{\text{target}} = \mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, 2r_{\text{ctr}})$. For $d = 1$, let $\bar{v}$ be the maximal velocity in $\mathcal{A}^1(r_{\text{cmm}}, 2r_{\text{ctr}})$ at $p = 0$, that is, $\bar{v} = \min\{r_{\text{cmm}}/m + (m-1)r_{\text{ctr}} \mid m \in \mathbb{N}\}$. Take three agents with positions $p^{[1]} = p^{[2]} = p^{[3]} = 0$ and velocities $v^{[1]} = -\bar{v}$, $v^{[2]} = 0$, and $v^{[3]} = \bar{v}$. At this configuration, the graph $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, 2r_{\text{ctr}})$ contains two edges: between agent 1 and 2 and between agent 2 and 3. Connectivity is preserved after one time step if agent 2 remains connected to both agents 1 and 3 after one time step. However, to remain connected with agent 1, its control $u^{[2]}$ must be equal to $-r_{\text{ctr}}$ and, analogously, to remain connected with agent 3, its control $u^{[2]}$ must be equal to $+r_{\text{ctr}}$. Clearly this is impossible. ■

Remarks 6.11 *(i) The result in Lemma 6.10 on the double integrator graph has the following interpretation. Assume that agent i has two neighbors j and k in the graph $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, r_{\text{ctr}})$. By definition of the neighboring law for this graph, we know that there exists bounded controls for i and j to maintain the $((p^{[i]}, v^{[i]}), (p^{[j]}, v^{[j]}))$ link and that there exists bounded controls for i and k to maintain the $((p^{[i]}, v^{[i]}), (p^{[k]}, v^{[k]}))$ link. The lemma states that, for some states of the agents i, j, and k, there might not exist controls that maintain both links simultaneously.*

(ii) *In other words, Lemma 6.10 shows how the disk graph $\mathcal{G}_{\text{disk}}(r_{\text{cmm}})$ and the double integrator disk graph $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, 2r_{\text{ctr}})$ are not appropriate choices for the connectivity maintenance problem.* □

The following theorem suggests that an appropriate scaling of the control bound is helpful in identifying a suitable state dependent graph for Problem 6.9.

Theorem 6.12 (A scaled double-integrator disk graph is suitable) *Consider a network of n agents with double integrator dynamics (6.9) in $\mathbb{R}^d$. Let r_{cmm} be the communication range and let r_{ctr} be the control bound. For $k \in \{1, \dots, n-1\}$, define*

$$\nu(k) = \frac{2}{k\sqrt{d}}.$$

Assume that $k \in \{1, \dots, n-1\}$ and the state $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$ together have the property that the graph $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k)r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$ contains a spanning tree T with diameter at most k. Then there exists $\{u^{[i]}\}_{i\in\{1,\dots,n\}} \subseteq B(0_d, r_{\text{ctr}})$ such that if $((p^{[i]}, v^{[i]}), (p^{[j]}, v^{[j]}))$ is an edge of T, then $((p^{[i]}+v^{[i]}, v^{[i]}+u^{[i]}), (p^{[j]}+v^{[j]}, v^{[j]}+u^{[j]}))$ is an edge of $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k)r_{\text{ctr}})$ at $\{(p^{[i]} + v^{[i]}, v^{[i]} + u^{[i]})\}_{i\in\{1,\dots,n\}}$.

These results are based upon Shostak's Theory for systems of inequalities, as exposed in [5]. We provide the proof in Appendix B. The following results are immediate consequences of this theorem.

Corollary 6.13 (Simple sufficient condition) *With the same notation in Theorem 6.12, define $\nu_{\min} = \frac{2}{(n-1)\sqrt{d}}$. Assume that the state $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$ is such that the graph $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu_{\min}r_{\text{ctr}})$ is connected at $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$. Then*

(i) *there exists $\{u^{[i]}\}_{i\in\{1,\dots,n\}} \subseteq B(0_d, r_{\text{ctr}})$, such that $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu_{\min}r_{\text{ctr}})$ is also connected at $\{(p^{[i]} + v^{[i]}, v^{[i]} + u^{[i]})\}_{i\in\{1,\dots,n\}}$; and*

(ii) *if T is a spanning tree of $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu_{\min}r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$, then there exists $\{u^{[i]}\}_{i\in\{1,\dots,n\}} \subseteq B(0_d, r_{\text{ctr}})$, such that, for all edges $((p^{[i]}, v^{[i]}), (p^{[j]}, v^{[j]}))$ of T, it holds that $((p^{[i]} + v^{[i]}, v^{[i]} + u^{[i]}), (p^{[j]} + v^{[j]}, v^{[j]} + u^{[j]}))$ is an edge of $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu_{\min}r_{\text{ctr}})$ at $\{(p^{[i]} + v^{[i]}, v^{[i]} + u^{[i]})\}_{i\in\{1,\dots,n\}}$.*

Remark 6.14 (Scaling of $\nu_{\min}$ with n) *The condition $\nu_{\min} = \frac{2}{\sqrt{d(n-1)}}$ implies that according to the sufficient conditions in Corollary 6.13, as the number of agents grows, the velocities of the agents must be closer and closer in order for the agents to be able to remain connected.*

If $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k) r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i \in \{1,\dots,n\}}$ is not connected for some k, then Theorem 6.12 applies to its connected components. In what follows we fix k and without loss of generality assume the graph $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k) r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i \in \{1,\dots,n\}}$ to be connected. □

Remark 6.15 (Distributed computation of connectivity) *Each agent can compute its neighbors in the graph $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k) r_{\text{ctr}})$ just by communicating with its neighbors in $\mathcal{G}_{\text{disk}}(r_{\text{cmm}})$ and exchanging with them position and velocity information. Alternatively, this computation may also be performed if each agent may sense relative position and velocity with all other agents at a distance no larger than r_{cmm}.* □

Remark 6.16 (Distributed computation of spanning trees) *It is possible to compute spanning trees in networks via standard depth-first search distributed algorithms. It is also possible [8] to distributively compute a minimum diameter spanning tree in a network.* □

6.3.4 The control constraint set

We now concentrate on two agents with indices i and j. For $t \in \mathbb{N}_0$, we define the relative position, velocity and control by $p^{[ij]}(t) = p^{[i]}(t) - p^{[j]}(t)$, $v^{[ij]}(t) = v^{[i]}(t) - v^{[j]}(t)$ and $u^{[ij]}(t) = u^{[i]}(t) - u^{[j]}(t)$, respectively. It is easy to see that

$$\begin{aligned} p^{[ij]}(t+1) &= p^{[ij]}(t) + v^{[ij]}(t), \\ v^{[ij]}(t+1) &= v^{[ij]}(t) + u^{[ij]}(t). \end{aligned}$$

Assume that agents i, j are connected in $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k) r_{\text{ctr}})$ at time t. By definition, this means that the relative state $(p^{[ij]}(t), v^{[ij]}(t))$ belongs to $\mathcal{A}^d(r_{\text{cmm}}, \nu(k) r_{\text{ctr}})$. If this connection is to be maintained at time $t+1$, then the relative control at time t must satisfy

$$u^{[i]}(t) - u^{[j]}(t) \in \mathcal{U}^d(r_{\text{cmm}}, \nu(k) r_{\text{ctr}}) \cdot (p^{[ij]}(t), v^{[ij]}(t)). \tag{6.10}$$

Also, implicit are the following bounds on individual control inputs $u^{[i]}(t)$ and $u^{[j]}(t)$:

$$u^{[i]}(t) \in B(0_d, r_{\text{ctr}}), \quad u^{[j]}(t) \in B(0_d, r_{\text{ctr}}). \tag{6.11}$$

This discussion motivates the following definition.

Definition 6.17 *Given $r_{\mathrm{cmm}}, r_{\mathrm{ctr}}, \nu(k) \in \mathbb{R}_+$ and given a set E of edges in $\mathcal{G}_{\text{di-disk}}(r_{\mathrm{cmm}}, \nu(k) r_{\mathrm{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$, the* control constraint set *is defined by*

$$\begin{aligned}
&\mathcal{U}^d_E(r_{\mathrm{cmm}}, r_{\mathrm{ctr}}, \nu(k)) \cdot (\{p^{[i]}, v^{[i]}\}_{i\in\{1,\dots,n\}}) \\
&\qquad = \{(u^{[1]},\dots,u^{[n]}) \in B(0_d, r_{\mathrm{ctr}})^n \mid \forall((p^{[i]}, v^{[i]}), (p^{[j]}, v^{[j]})) \in E, \\
&\qquad\qquad u^{[i]} - u^{[j]} \in \mathcal{U}^d(r_{\mathrm{cmm}}, \nu(k) r_{\mathrm{ctr}}) \cdot (p^{[i]} - p^{[j]}, v^{[i]} - v^{[j]})\}.
\end{aligned}$$

In other words, the control constraint set for an edge set E is the set of controls for each agent with the property that all edges in E will be maintained in one time step.

Remark 6.18 *We can now interpret the results in Theorem 6.12 as follows.*

(i) To maintain connectivity between any pair of connected agents, we should simultaneously handle constraints corresponding to all *edges of $\mathcal{G}_{\text{di-disk}}(r_{\mathrm{cmm}}, \nu(k) r_{\mathrm{ctr}})$. This might render the control constraint set empty.*

(ii) However, if we only consider constraints corresponding to edges belonging to a *spanning tree T of $\mathcal{G}_{\text{di-disk}}(r_{\mathrm{cmm}}, \nu(k) r_{\mathrm{ctr}})$, then the set $\mathcal{U}^d_T(r_{\mathrm{cmm}}, \nu(k) r_{\mathrm{ctr}}) \cdot (\{p^{[i]}, v^{[i]}\}_{i\in\{1,\dots,n\}})$ is guaranteed to be nonempty.* □

Let us now provide a concrete representation of the control constraint set. Given a pair i, j of connected agents, the admissible control set $\mathcal{U}^d(r_{\mathrm{cmm}}, \nu(k) r_{\mathrm{ctr}}) \cdot (p^{[ij]}, v^{[ij]})$ is convex and compact (Lemma 6.2). Hence, we can fit a polytope with N_{poly} sides inside it. This approximating polytope leads to the following tighter version of the constraint in (6.10):

$$(C^\eta_{ij})^T (u^{[i]} - u^{[j]}) \le w^\eta_{ij}, \qquad \eta \in \{1,\dots,N_{\mathrm{poly}}\}, \tag{6.12}$$

for some appropriate vector $C^\eta_{ij} \in \mathbb{R}^d$ and scalar $w^\eta_{ij} \in \mathbb{R}$. Similarly, one can compute an inner polytopic approximation of the closed ball $B(0_d, r_{\mathrm{ctr}})$ and write the following linear vector inequalities:

$$(C^\eta_{i\theta})^T u^{[i]} \le w^\eta_{i\theta}, \qquad \eta \in \{1,\dots,N_{\mathrm{poly}}\}, \tag{6.13}$$

where the symbol θ has the interpretation of a fictional agent. In this way, we have cast the original problem of finding a set of feasible control inputs into a satisfiability problem for a set of linear inequalities.

Remark 6.19 *Rather than a network-wide control constraint set, one might like to obtain decoupled constraint sets for each individual agent. However, (1) it is not clear how to design a distributed algorithm to perform this computation, (2) such an algorithm will likely have large communication requirements, and (3) such a calculation might lead to a very conservative estimate for the decoupled control constraint sets. Therefore, rather than explicitly decoupling the control constraint sets, we next focus on a distributed algorithm to search the control constraint set for feasible controls that are optimal according to some criterion.* □

6.4 Distributed control computation

In this section we formulate and solve the following optimization problem: given an array of desired control inputs $U_{\text{des}} = (u_{\text{des}}^{[1]}, \dots, u_{\text{des}}^{[n]})^T \in (\mathbb{R}^d)^n$, find, via local computation, the array $U = (u^{[1]}, \dots, u^{[n]})$ belonging to the control constraint set, that is *closest* to the desired array U_{des}. To formulate this problem precisely, we need some additional notations. Let E be a set of edges in the undirected graph $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k) r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i \in \{1,\dots,n\}}$. To deal with the linear inequalities of the form (6.12) and (6.13) associated to each edge of E, we introduce an appropriate *multigraph*. A *multigraph* (or *multiple edge graph*) is, roughly speaking, a graph with multiple edges between the same vertices. More formally, a multigraph is a pair $(V_{\text{mult}}, E_{\text{mult}})$, where V_{mult} is the vertex set and the edge set E_{mult} contains numbered edges of the form (i, j, η), for $i, j \in V$ and $\eta \in \mathbb{N}$, and where E_{mult} has the property that if $(i, j, \eta) \in E_{\text{mult}}$ and $\eta > 1$, then also $(i, j, \eta - 1) \in E_{\text{mult}}$. In what follows, we let G_{mult} denote the multigraph with vertex set $\{1, \dots, n\}$ and with edge set $E_{\text{mult}} = \{(i, j, \eta) \in \{1, \dots, n\}^2 \times \{1, \dots, N_{\text{poly}}\} \mid ((p^{[i]}, v^{[i]}), (p^{[j]}, v^{[j]})) \in E,\ i > j\}$. Note that to each element $(i, j, \eta) \in E_{\text{mult}}$ is associated the inequality $(C_{ij}^{\eta})^T (u^{[i]} - u^{[j]}) \leq w_{ij}^{\eta}$. We are now ready to formally state the optimization problem at hand in the form of the following quadratic programming problem:

$$
\begin{aligned}
\text{minimize} \quad & \frac{1}{2} \sum_{i=1}^{n} \| u^{[i]} - u_{\text{des}}^{[i]} \|^2, \\
\text{subj. to} \quad & (C_{ij}^{\eta})^T (u^{[i]} - u^{[j]}) \leq w_{ij}^{\eta}, \quad \text{for } (i, j, \eta) \in E_{\text{mult}}, \\
& (C_{i\theta}^{\eta})^T u^{[i]} \leq w_{i\theta}^{\eta}, \qquad \text{for } i \in \{1, \dots, n\}, \eta \in \{1, \dots, N_{\text{poly}}\}.
\end{aligned}
\tag{6.14}
$$

Here, somehow arbitrarily, we have adopted the 2-norm to define the cost function.

Remark 6.20 (Feasibility) *If E is a spanning tree of $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu r_{\text{ctr}})$ at a connected configuration $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$, then the control constraint set $\mathcal{U}_E^d(r_{\text{cmm}}, r_{\text{ctr}}, \nu(k)) \cdot (\{p^{[i]}, v^{[i]}\}_{i\in\{1,\dots,n\}})$ is guaranteed to be non-empty by Theorem 6.12. In turn, this implies that the optimization problem* (6.14) *is feasible.* □

6.4.1 Solution via duality: the projected Jacobi method

The problem (6.14) can be written in a compact form as:

$$\begin{aligned} &\text{minimize } \frac{1}{2}\|U - U_{\text{des}}\|^2, \\ &\text{subj. to } B_{\text{mult}}^T U \preceq w, \end{aligned}$$

for appropriately defined matrix B_{mult} and vector w. A dual "projected Jacobi method" algorithm for the solution of this standard quadratic program is described in Appendix A. According to this algorithm, let λ^* be the value of Lagrange multipliers at optimality. Then the global minimum for U is achieved at

$$U^* = U_{\text{des}} - B_{\text{mult}}\lambda^*. \tag{6.15}$$

The projected Jacobi iteration for each component of λ is given by

$$\begin{aligned} \lambda_\alpha(t+1) = \max\Big\{0, \lambda_\alpha(t) - \frac{\tau}{(B_{\text{mult}}^T B_{\text{mult}})_{\alpha\alpha}} \Big((w - B_{\text{mult}}^T U_{\text{des}})_\alpha \\ + \sum_{\beta=1}^{N_{\text{poly}}(e+n)} (B_{\text{mult}}^T B_{\text{mult}})_{\alpha\beta}\lambda_\beta(t) \Big) \Big\}, \end{aligned} \tag{6.16}$$

where $\alpha \in \{1, \dots, N_{\text{poly}}(e+n)\}$ and τ is the step size parameter. We can select $\tau = \frac{1}{N_{\text{poly}}(e+n)}$ to guarantee convergence.

6.4.2 A distributed dual algorithm

Because of the particular structure of the matrix $B_{\text{mult}}^T B_{\text{mult}}$, the iterations (6.16) can be implemented in a distributed way over the original graph G.

To highlight the distributed structure of the iteration we denote the components of λ by referring to the nodes that they share and the inequality they are related to. In particular for each edge in G_{mult}, the corresponding Lagrange multiplier will be denoted as λ_{ij}^{η} if the edge goes from node i to node j, $i > j$, and the edge is associated to the inequality constraint $C_{ij}^{\eta}(u^{[i]} - u^{[j]}) \leq w_{ij}^{\eta}$. This makes up the first $N_{\text{poly}}e$ entries of the vector λ. To be consistent with this notation, the next $N_{\text{poly}}n$ entries will be denoted $\lambda_{1\theta}^{1}, \dots, \lambda_{1\theta}^{N_{\text{poly}}}, \dots, \lambda_{n\theta}^{1}, \dots, \lambda_{n\theta}^{N_{\text{poly}}}$. Additionally, define $\mathcal{N}(i) = \{j \in \{1, \dots, n\} \mid \{(p^{[i]}, v^{[i]}), (p^{[j]}, v^{[j]})\} \in E\} \cup \{\theta\}$. The symbol θ has the interpretation of a fictional node.

Defining $\lambda_{ij}^{\eta} := \lambda_{ji}^{\eta}$ and $C_{ij}^{\eta} := -C_{ji}^{\eta}$ for $i < j$, we can write equations (6.15) and (6.16) in components as follows. Equation (6.15) reads, for $i \in \{1, \dots, n\}$,

$$u^{[i]*} = u_{\text{des}}^{[i]} - \sum_{k \in \mathcal{N}(i)} \sum_{\eta=1}^{N_{\text{poly}}} C_{ik}^{\eta} \lambda_{ik}^{\eta}.$$

One can easily work an explicit expression for matrix product $B_{\text{mult}}^{T} B_{\text{mult}}$ in (6.16). Then, equation (6.16) reads, for $(i, j, \eta) \in E_{\text{mult}}$,

$$\begin{aligned}\lambda_{ij}^{\eta}(t+1) = \max \Bigg\{ 0, \lambda_{ij}^{\eta}(t) - \frac{\tau}{2(C_{ij}^{\eta})^{T} C_{ij}^{\eta}} \cdot \\ \Bigg(\sum_{k \in \mathcal{N}(i)} \sum_{\sigma=1}^{N_{\text{poly}}} \Big((C_{ij}^{\eta})^{T} C_{ik}^{\sigma} \lambda_{ik}^{\sigma} \Big) + \sum_{k \in \mathcal{N}(j)} \sum_{\sigma=1}^{N_{\text{poly}}} \Big((C_{ji}^{\eta})^{T} C_{jk}^{\sigma} \lambda_{jk}^{\sigma} \Big) \\ + w_{ij}^{\eta} - (C_{ij}^{\eta})^{T} (u_{\text{des}}^{[i]} - u_{\text{des}}^{[j]}) \Bigg) \Bigg\},\end{aligned}$$

together with, for $i \in \{1, \dots, n\}$, $\eta \in \{1, \dots, N_{\text{poly}}\}$,

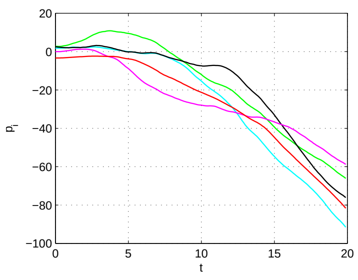
$$\begin{aligned}\lambda_{i\theta}^{\eta}(t+1) = \max \Big\{ 0, \lambda_{i\theta}^{\eta}(t) \\ - \frac{\tau}{(C_{i\theta}^{\eta})^{T} C_{i\theta}^{\eta}} \Big(\sum_{k \in \mathcal{N}(i)} \sum_{\sigma=1}^{N_{\text{poly}}} ((C_{i\theta}^{\eta})^{T} C_{ik}^{\sigma} \lambda_{ik}^{\sigma}) + w_{i\theta}^{\eta} - (C_{i\theta}^{\eta})^{T} u_{\text{des}}^{[i]} \Big) \Big\}.\end{aligned}$$

We distribute the task of running iterations for these $N_{\text{poly}}(e+n)$ Lagrange multipliers among the n agents as follows: an agent i carries out the updates for all quantities $\lambda_{i\theta}^{\eta}$ and all λ_{ij}^{η} for which $i > j$. By means of this partition and by means of iterated one-hop communication among agents, it is possible to compute the global solution for the optimization problem (6.14) in a distributed fashion over the double integrator disk graph.

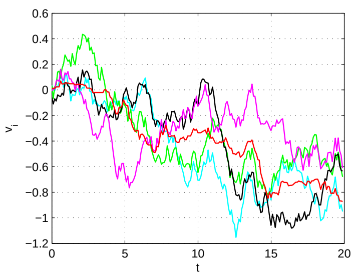
6.5 Simulations

To illustrate our analysis we focus on the following scenario. For the two dimensional setting, i.e., for $d = 2$, we assume that there are $n = 5$ agents with (randomly chosen) initial condition and such that they are connected according to $\mathcal{G}_{\text{di-disk}}$. The bound for the control input is $r_{\text{ctr}} = 2$ and the communication radius is $r_{\text{cmm}} = 10$.

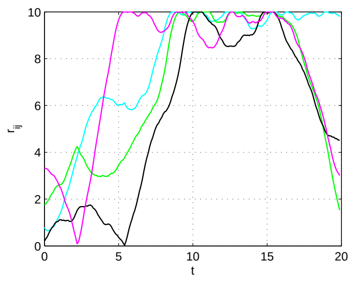
In the first scenario we assign to each agent a random value, chosen from a normal distribution with zero mean and r_{ctr}^2 variance, at each time step as desired control. In Figures 6.4a and 6.4b the positions and velocities of the agents with respect to time are plotted. In Figure 6.4c the distances between agents which are neighbors in the spanning tree are shown. The distances are always less than $r_{\text{cmm}} = 10$, which means that the graph remains connected.



(a) Positions



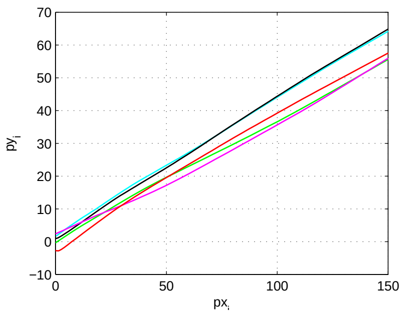
(b) Velocities (v_x and v_y)



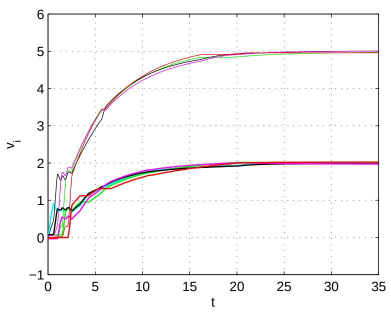
(c) Inter-agent distances

Figure 6.4: Random connected motion for 5 agents in the plane ($d = 2$)

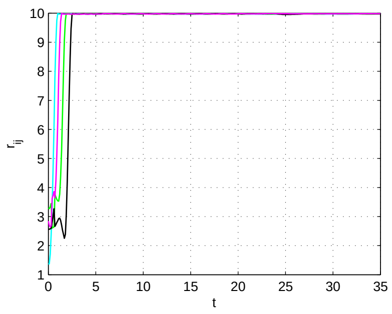
In the second scenario we assigned to one of the agents a derivative feedback control $u_x(p, v) = (v_x - 2)$, $u_y(p, v) = (v_y - 5)$ as desired input. For the other agents the desired input is set to zero. We show the agent trajectories in Figure 6.5a, the velocities of the agents with respect to time in Figure 6.5b, and the distances between agents which are neighbors in the spanning tree in Figure 6.5c. Notice that the agents move with approximately identical velocity reaching a configuration in which all of them are at the limit distance $r_{\text{cmm}} = 10$. The interesting aspect of this simulation is that the maintenance of connectivity leads to the accomplishment of apparently unrelated coordination tasks as velocity alignment and cohesiveness. This numerical result illustrate how our connectivity maintenance approach might indeed be a starting point for novel investigations into the problem of flocking with connectivity.



(a) Positions



(b) Velocities (v_x and v_y)



(c) Inter-agent distances

Figure 6.5: Velocity alignment and cohesiveness for 5 agents in the plane ($d = 2$)

6.6 Discussion

We provide distributed algorithms to enforce connectivity among networks of agents with double-integrator dynamics. Future directions of research include (i) evaluating the communication complexity of the proposed distributed dual algorithm and possibly designing faster ones, (ii) studying the relationship between the connectivity maintenance problem and the platooning and mesh stability problem, and (iii) investigating the flocking phenomenon and designing flocking algorithms which do not rely on a blanket assumption of connectivity.

Chapter 7

Network abstract linear programming

In this chapter we identify a novel class of distributed optimization problems, namely a networked version of abstract linear programming. For such problems we propose distributed algorithms for networks with various connectivity and/or memory constraints.

7.1 Introduction

This chapter focuses on a class of distributed optimization problems. We study abstract linear programming, that is, a generalized version of linear programming that was introduced by Matousěk, Sharir and Welzl in [49] and extended by Gärtner in [21]. Abstract linear programming is applicable also to some geometric optimization problems, such as the minimum enclosing ball, the minimum enclosing stripe and the minimum enclosing annulus. These geometric optimization problems are relevant in the design of efficient robotic algorithms for minimum-time formation control problems.

Linear programming and its generalizations have received widespread attention in the literature. The following references are most relevant in our treatment. The earliest (deterministic) algorithm that solves a linear program in a fixed number of variables subject to n linear inequalities in time $O(n)$ is given in [51]. An efficient randomized incremental algorithm for linear programming is proposed in [49], where a linear program in d variables subject to n linear inequalities is solved in expected time $O(d^2n+e^{O(\sqrt{d}\log d)})$; the expectation is taken over the internal randomizations executed by the algorithm. An elegant survey on randomized methods in linear programming is [22]. The survey [1] discusses the application of abstract linear programming to a number of geometric optimization problems. Regarding parallel computation approaches to linear programming, we only note that linear programs with n linear inequalities can be solved [2] by n parallel processors in time $O((\log\log(n))^d)$. The approach in [2] and the ones in the references therein are, however, limited to parallel random-access machines (usually denoted PRAM), where a shared memory is readable and writable to all processors. In this chapter, we focus on networks described by arbitrary graphs and on robotic networks described by geometric graphs.

The contributions of this chapter are two-fold. First, we identify a class of distributed optimization problems that appears to be novel and of intrinsic interest. Basically, we show how to formulate an abstract linear program over a network of processors. Second, we propose a novel simple algorithmic methodology to solve these problems in networks with various connectivity and/or memory constraints. Specifically, we propose three algorithms. The first algorithm is suitable for time-dependent networks, whose nodes have bounded in-degree that has to be designed in order to deal with bounded computation time between two communication rounds. The second algorithm, for time-dependent networks, manages bounded computation time while allowing free in-degree. However it needs arbitrarily large memory (depending on the maximum in-degree). Finally, the third algorithm manages both bounded computation time and bounded memory with arbitrarily large in-degree, but works only on time-independent networks. We prove correctness of the algorithms and establish halting conditions.

The chapter is organized as follows. Section 7.2 introduces abstract linear programs after a short review on linear programming. Section 7.3 contains the definition of network abstract linear programs and the proposed distributed algorithms. Finally, in Section 7.4 we draw conclusions and suggest future perspectives.

7.2 Abstract linear programming

In this section we present an abstract framework that captures a wide class of optimization problems including linear programming and various geometric optimization problems. These problems are known as *abstract linear programs* (or *LP-type problems*). They can be considered a generalization of linear programming in the sense that they share some important properties.

A comprehensive analysis of these problems may be found for example in [1].

7.2.1 Linear Programming

Before describing this abstract framework, we revise main features of Linear Programming. LP problems are probably the most studied optimization problems.

We refer to the linear programming problem in the following standard form. Given variables $x = (x_1, \dots, x_d)$ and constraints $a_i x \le b_i$, $i \in \{1, \dots, n\}$, find x which maximizes the value $\{c{\cdot}x | Ax \le b\}$. Geometrically, this is equivalent to finding a vertex x^* extreme in some direction ϕ within the polyhedron P defined by the intersection of a set H of n closed halfspaces in $\mathbb{R}^d$. Figure 7.1 provides an example. An good reference for linear programming and, in general, convex programming is [7].

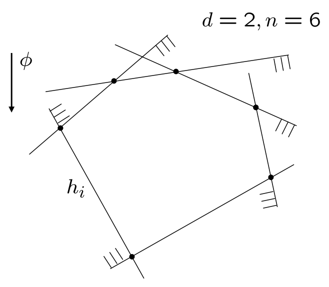


Figure 7.1: Geometric linear program

7.2.2 Abstract framework

Let us consider optimization problems specified by a pair (H, ω), where H is a finite set, and $\omega : 2^H \to \Omega$ is a function with values in a linearly ordered set $(\Omega, \le)$; we assume that Ω has a minimum value $-\infty$. The elements of H are called *constraints*, and for $G \subset H$, $\omega(G)$ is called the *value* of G. Intuitively, $\omega(G)$ is the smallest value attainable by a certain objective function while satisfying the constraints of G. An optimization problem of this sort is called *abstract linear program* if the following axioms are satisfied:

(i) *Monotonicity*: if $F \subset G \subset H$, then

$$\omega(F) \le \omega(G);$$

(ii) *Locality*: if $F \subset G \subset H$ with $-\infty < \omega(F) = \omega(G)$, then, for all $h \in H$,

$$\omega(G) < \omega(G \cup \{h\}) \quad \Longrightarrow \quad w(F) < w(F \cup \{h\}).$$

A set $B \subset H$ is *minimal* if $\omega(B) > \omega(B')$ for all proper subsets B' of B. A minimal set B with $-\infty < \omega(B)$ is a *basis*. Given $G \subset H$, a *basis of* G is a minimal subset of constraints $B \subset G$, such that $-\infty < \omega(B) = \omega(G)$. A constrained h is said to be *violated* by G, if $\omega(G) < \omega(G \cup \{h\})$. A constraint h is *extreme* in G if $\omega(G) < \omega(G \setminus \{h\})$.

The *solution* of an abstract linear program (H, ω) is a minimal set $B_H \subset H$ with the property that $\omega(B_H) = \omega(H)$. The *combinatorial dimension* δ of (H, ω) is the maximum cardinality of any basis. Finally, an abstract linear program is called *basis regular* if for any basis with $\text{card}(B) = \delta$ and any constraint $h \in H$, every basis of $B \cup \{h\}$ has the same cardinality of B. We now define two important primitive operations that are useful in the solution of the abstract linear program:

(i) *Violation test*: given a constraint h and a basis B, it tests whether h is violated by B; we denote this primitive by $\texttt{Viol}(B, h)$;

(ii) *Basis computation*: given a constraint h and a basis B, it computes a basis of $B \cup \{h\}$. we denote this primitive by $\texttt{Basis}(B, h)$.

Remark 7.1 (Examples of abstract linear programs) *We present three geometric examples that will be useful later in the chapter.*

(i) Smallest enclosing ball: *Given n points in $\mathbb{R}^d$, compute the center and radius of the ball of smallest volume containing all the points. This problem has combinatorial dimension* $d + 1$.

(ii) Smallest enclosing stripe: *Given n points in $\mathbb{R}^2$ in* generic positions[1]*, compute the center and the width of the stripe of smallest width containing all the points. This problem has combinatorial dimension* 5.

(iii) Smallest enclosing annulus: *Given n points in $\mathbb{R}^2$, compute the center and the two radiuses of the annulus of smallest area containing all the points. This problem has combinatorial dimension* 4.

These three problems are illustrated in Figure 7.2. Numerous other geomet-

[1]The notion of generic positions will be clarified in the next chapter

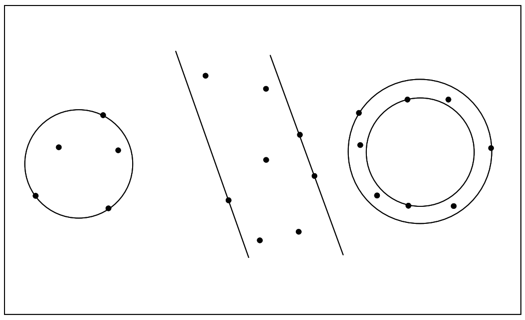


Figure 7.2: Smallest enclosing ball, stripe and annulus

ric optimization problems can be cast as abstract linear programs. Examples include computing the smallest enclosing ellipsoid, the largest ellipsoid in a polytope, the smallest enclosing orthotope, the distance between convex polytopes and others. More examples are discussed in [49, 21, 22, 1] and references therein. □

Remark 7.2 *A surprising feature of abstract linear programs is that some of them are nonlinear and non-convex programs; see [1].* □

7.2.3 Randomized sub-exponential algorithm

A randomized algorithm for solving abstract linear programs has been proposed in [49]. Such algorithm has linear expected running time in terms of the number of constraints whenever the combinatorial dimension δ is fixed and subexponential in δ. The model of computation used to determine the complexity is the *real RAM*, which is widely used in computational geometry: each arithmetic operation with real numbers (the allowed inputs of the algorithm) is charged unit cost. Sometimes this is also referred to as *combinatorial complexity.*

The algorithm, called `SUBEX_lp`, has a recursive structure and is based on the two primitives introduced above, i.e., the violation test and the basis computation primitives. For simplicity, we assume here that such primitives may be implemented in time $\Theta(1)$ with respect to the number of constraints. Given a set of constraints G and a candidate basis $C \subset G$, the recursive algorithm is as follows.

```
function
SUBEX_lp($G, C$)
  if $G = C$, then
    return $C$
  else
    choose a random $h \in G \setminus C$
    $B :=$ SUBEX_lp($G \setminus \{h\}, C$)
    if Viol($B, h$), i.e., $h$ is violated by $B$, then
      return
      SUBEX_lp($G$, Basis($B, h$))
    else
      return $B$
    end if
  end if
```

For the abstract linear program (H, ω), the algorithm is invoked with

$$\texttt{SUBEX\_lp}(H, B),$$

given any initial candidate basis B.

In [49] the expected completion time for the `SUBEX_lp` algorithm was shown to be in $O(d^2 n + e^{O(\sqrt{d \log d})})$ for basis regular abstract linear programs. In [22] the result was extended to problems that are not basis regular.

7.3 Network abstract linear programming

In this section we define a *network abstract linear program* and propose novel distributed algorithms to solve it.

7.3.1 Problem statement

Informally we can say that a *network abstract linear program* consists of three main elements: a network, an abstract linear program and a mapping that associates to each constraint of the abstract linear program a node of the network. A more formal definition is the following.

Definition 7.3 *A network abstract linear program (NALP) is a tuple* $(\mathcal{G}, (H, \omega), \mathcal{B})$ *consisting of* *(i)* $\mathcal{G} = (I, E)$*, a communication digraph;*

(ii) (H, ω)*, an abstract linear program;*

(iii) $\mathcal{B} : H \to I$*, a surjective map called* constraint distribution map. □

The *solution* of the network abstract linear program is attained when all processor in the network have computed a solution to the abstract linear program.

Remark 7.4 *Our definition allows for various versions of network abstract linear programs. Regarding the constraint distribution map, the most natural case to consider is when the constraint distribution map is bijective. In this case one constraint is assigned to each node. A similar situation occurs when multiple constraints are assigned to each node. More complex distribution laws are also interesting depending on the computation power and memory of the processors in the network. In what follows, we assume* $\mathcal{B}$ *to be bijective.* □

7.3.2 Distributed algorithms

Next we define three distributed algorithms that solve network abstract linear programs. First, we describe a synchronous version that is well suited for time-dependent networks whose nodes have bounded computation time and memory, but also bounded in-degree or equivalently arbitrary in-degree, but also arbitrary computation time and memory. Then we describe two variations that take into account the problem of dealing with arbitrary in-degree versus short computation time and small memory. The second version of the algorithm is suited for time-dependent networks that have arbitrary in-degree and bounded computation time, but are allowed to store arbitrarily large amount of information, in the sense that the number of stored messages may depend on the number of nodes of the network. The third algorithm considers the case of time-independent networks with arbitrary in-degree and bounded computation time and memory.

In the algorithms we consider a uniform network $\mathcal{S}$ with communication digraph $\mathcal{G} = (I, E_{\text{cmm}})$ and a network abstract linear program $(\mathcal{G}, (H, \omega), \mathcal{B})$. We assume $\mathcal{B}$ to be bijective, that is, the set of constraints H has dimension n, $H = \{h_1, \cdots, h_n\}$. The combinatorial dimension is δ.

Here is an informal description of what we shall refer to as the *FloodBasis* algorithm:

[Informal description] Each process has a logical state of $\delta + 1$ variables taking values in H. The first δ components represent the current value of the basis to compute, while the last element is the constraint assigned to that node. At the start round the process initializes every component of the basis to its constraint, then, at each communication round, performs the following tasks: (i) it acquires from its neighbors (a message consisting of) their current basis; (ii) it executes the `SUBEX_lp` algorithm over the constraint set given by the collection of its and its neighbors' basis and its constraint (that it maintains in memory), thus computing a new basis; (iii) it updates its logical state and message using the new basis obtained in (ii).

The algorithm is described formally in the table.

`Distributed algorithm:`	*FloodBasis*
`Goal:`	Solve NALP
`Message alphabet:`	$M = H \cup \{\texttt{null}\}$
`Logical state:`	$w^{[i]} = (B^{[i]}, h^{[i]})$
	$B^{[i]} \in H^{\delta}$, $h^{[i]} \in H$
`Initialization:`	$B^{[i]} = (h^{[i]}, \cdots, h^{[i]})$, $h^{[i]} = h_i$

```
function msg($w^{[i]}, j$)
 1: return $B^{[i]}$
function stf($w^{[i]}, y$)
 1: collect $y^{[i]} := \{\text{msg}(w^{[j]}, i) \mid j \in \mathcal{N}_I(i)\}$
 2: collect $H_i := (y^{[i]}, w^{[i]})$
 3: compute $B^{[i]} :=$ SUBEX_lp($H_i, B^{[i]}$)
 4: return $(B^{[i]}, h^{[i]})$
```

In the second scenario we work with a time-dependent network with no bounds on the in-degree of the nodes and on the memory size. In this setting the execution of the `SUBEX_lp` may exceed the communication round length. In order to deal with this problem, we slightly change the network model as described in Section 5.3 of Chapter 5, so that each process may execute the state transition function "asynchronously", in the sense that the time-length of the execution may take multiple rounds. If that happens, the message generation function in each intermediate round is called using the logical state of the previous round. Here is an informal description of what we shall refer to as the *FloodBasisMultiRound* algorithm:

> *[Informal description]* Each process has the same message alphabet and logical state as in *FloodBasis* and also the same state initialization. At each communication round it performs the following tasks: i) it acquires the messages from its in-neighbors; ii) if the execution of the `SUBEX_lp` at the previous round was over it starts a new instance, otherwise it keeps executing the one in progress; iii) if the execution of the `SUBEX_lp` ends it updates the logical state and runs the message-generation function with the new state, otherwise it generates the same message as in the previous round.

As regards the sets and functions, the second algorithm has exactly the same message alphabet, logical state, message function and state transition function as the *FloodBasis*, therefore the formal scheme is the same as in the *FloodBasis* table with the remark on the process structure said above.

In the third scenario we work with a time-independent network with no bounds on the in-degree of the nodes. We suppose that each processor has limited memory capacity, so that it can store at most D messages. The memory is dimensioned so to guarantee that the `SUBEX_lp` is always solvable during two communication rounds. The memory constraint is solved by processing only part of the incoming messages at each round and cycling in a suitable way in order to process all the messages in multiple rounds.

Here is an informal description of what we shall refer to as the *FloodBasisCycling* algorithm:

> *[Informal description]* The first $\delta + 1$ components of the logical state are the same as in *FloodBasis* and are initialized in the same way. A further component is added. It is simply a counter variable that keeps trace of the current round. At each communication round each process performs the following tasks: (i) it acquires from its neighbors (a message consisting of) their current basis; (ii) it chooses D messages according to a scheduled protocol, e.g. it labels its in-neighboring edges with natural numbers from 1 up to $\texttt{indeg}^{[i]}$ and cycles over them in increasing order; (iii) it executes the `SUBEX_lp` algorithm over the constraint set given by the collection of the D messages plus its basis and its

constraint (that it maintains in memory), thus computing a new basis; (iv) it updates its logical state and message using the new basis obtained in (iii).

The algorithm is described formally in the table.

`Distributed algorithm:`	*FloodBasisCycling*
`Goal:`	Solve NALP with bounded memory and computation time
`Message alphabet:`	$M = H \cup \{\texttt{null}\}$
`Logical state:`	$w^{[i]} = (B^{[i]}, h^{[i]}, \texttt{round}^{[i]})$
	$B^{[i]} \in H^{\delta}$, $h^{[i]} \in H$, $\texttt{round}^{[i]} \in \mathbb{N}_0$
`Initialization:`	$B^{[i]} := (h^{[i]}, \cdots, h^{[i]})$, $h^{[i]} = h_i$, $\texttt{round}^{[i]} := 0$

```
function msg($w^{[i]}, j$)
  1: return $B^{[i]}$
function stf($w^{[i]}, y$)
  1: define $y^{[i]} := \{\mathrm{msg}(w^{[j]}, i) \mid j \in \mathcal{N}_I(i)\}$
  2: compute $\texttt{roundMOD} := \texttt{round}^{[i]} \mod \left\lceil \frac{\texttt{indeg}^{[i]}}{D} \right\rceil$
  3: set $k := D\,\texttt{roundMOD}$
  4: collect $H_i := \left(y^{[i]}_{(k+1)}, \cdots, y^{[i]}_{\min\{\texttt{indeg}^{[i]}-1, k+D\}}, w^{[i]}\right)$
  5: compute $B^{[i]} := \texttt{SUBEX\_lp}(H_i, B^{[i]})$
  6: return $(B^{[i]}, h^{[i]}, \texttt{round}^{[i]} + 1)$
```

Remark 7.5 *For the algorithm to converge it is important that each agent keeps in memory its constraint and thus implements the* `SUBEX_lp` *on the bases received from its neighbors together with its constraint. This requirement is important because of the following reason: no element of a basis B for a set $G \subset H$ needs to be an element in the basis of $G \cup \{h\}$ for any $h \in H \setminus G$.* □

We are now ready to prove the algorithms' correctness.

Proposition 7.6 (Correctness of *FloodBasis*) *Let $\mathcal{S}$ be a synchronous time-dependent network with communication digraph $\mathcal{G} = (I, E_{\mathrm{cmm}})$ and let*

$(\mathcal{G},(H,\omega),\mathcal{B})$ *be a network abstract linear program. If* $\mathcal{G}$ *is jointly strongly connected, then the* FloodBasis *algorithm solves* $(\mathcal{G},(H,\omega),\mathcal{B})$*, that is, in a finite number of communication rounds each node acquires a copy of the solution of* (H,ω)*, i.e., the basis* B *of* H*.* □

Proof: In order to prove correctness of the algorithm, observe, first of all, that each law at every node converges in a finite number of steps. In fact, using axioms from abstract linear programming and finiteness of H, each sequence $\omega(B^{[i]}(t))$, $t \in \mathbb{N}$, is monotone nondecreasing, upper bounded and can assume a finite number of values. Then we proceed by contradiction to prove that all the laws converge to the same $\omega(B)$ and that it is exactly $\omega(B) = \omega(H)$. Suppose that for $t > t_0 > 0$ all the nodes have converged to their limit basis and that there exist at least two nodes, call them i and j, such that $\omega(B^{[i]}(t)) = \omega(B^{[i]}) \neq \omega(B^{[j]}) = \omega(B^{[j]}(t))$, for all $t \geq t_0$. For $t = t_0 + 1$, for every $k_1 \in \mathcal{N}_O(i)$, $B^{[i]}$ does not violate $B^{[k_1]}$, otherwise they would compute a new basis thus violating the assumption that they have converged. Using the same argument at $t = t_0 + 2$, for every $k_2 \in \mathcal{N}_O(k_1)$, $B^{[k_1]}$ does not violate $B^{[k_2]}$. Notice that this does not imply that $B^{[i]}$ does not violate $B^{[k_2]}$, but it implies that $\omega(B^{[i]}) \leq \omega(B^{[k_2]})$. Iterating this argument we can show that for every $S > 0$, every k connected to i in the graph $\cup_{t=t_0}^{t_0+S}\mathcal{G}(t)$ must have a basis $B^{[k]}$ such that $\omega(B^{[i]}) \leq \omega(B^{[k]})$. However, using the joint connectivity assumption, there exists $S_0 > 0$ such that $\cup_{t=t_0}^{t_0+S_0}\mathcal{G}(t)$ is strongly connected and therefore i is connected to j, thus showing that $\omega(B^{[i]}) \leq \omega(B^{[j]})$. Repeating the same argument by starting from node j we obtain that $\omega(B^{[j]}) \leq \omega(B^{[i]})$, that implies $\omega(B^{[i]}) = \omega(B^{[j]})$, thus giving the contradiction. Now, the basis at each node satisfies, by construction, the constraints of that node. Since the basis is the same for each node, it satisfies all the constraints, then $\omega(B) = \omega(H)$. ■

Remark 7.7 *Correctness of the other two versions of the* FloodBasis *algorithm may be established along the same lines. For example, it is immediate to establish that the basis at each node reaches a constant value in finite time. It is easy to show that this constant value is the solution of the abstract linear program for the* FloodBasisMultiRound *algorithm. For the* FloodBasisCycling *algorithm we note that the procedure used to process the incoming data is equivalent to considering a time-dependent graph whose edges change with that law.* □

Proposition 7.8 (Halting condition) *Consider a network* $\mathcal{S}$ *with time-independent, strongly connected digraph* $\mathcal{G}$ *where the* FloodBasis *algorithm* *is running. Each process can halt the algorithm execution if the value of its basis has not changed after* $2\,\mathrm{diam}(\mathcal{G})+1$ *communication rounds.* □

Proof: First, notice that, for all $t \in \mathbb{N}_0$ and for every $(i,j) \in E_{\mathrm{cmm}}$,

$$\omega(B^{[i]}(t)) \leq \omega(B^{[j]}(t+1)). \tag{7.1}$$

This holds by simply noting that $B^{[j]}(t+1)$ is not violated by $B^{[i]}(t)$ by construction of the *FloodBasis* algorithm. Assume that node i satisfies $B^{[i]}(t) = B$ for all $t \in \{t_0, \dots, t_0+2\,\mathrm{diam}(\mathcal{G})\}$, and pick any other node j. Without loss of generality assume that $t_0 = 0$. Because of equation (7.1), if $k_1 \in \mathcal{N}_O(i)$, then $\omega(B^{[k_1]}(1)) \geq \omega(B)$ and, recursively, if $k_2 \in \mathcal{N}_O(k_1)$, then $\omega(B^{[k_2]}(2)) \geq \omega(B^{[k_1]}(1)) \geq \omega(B)$. Iterating this argument $\mathrm{dist}(i,j)$ times, the node j satisfies $\omega(B^{[j]}(\mathrm{dist}(i,j))) \geq \omega(B)$. Now, consider the out-neighbors of node j. For every $k_3 \in \mathcal{N}_O(j)$, it must hold that $\omega(B^{[k_3]}(\mathrm{dist}(i,j)+1)) \geq \omega(B^{[j]}(t))$. Iterating this argument $\mathrm{dist}(j,i)$ times, the node i satisfies $\omega(B^{[i]}(\mathrm{dist}(i,j) + \mathrm{dist}(j,i))) \geq \omega(B^{[j]}(\mathrm{dist}(i,j)))$. In summary, because $\mathrm{dist}(i,j) + \mathrm{dist}(j,i) \leq 2\,\mathrm{diam}(\mathcal{G})$, we know that $B^{[i]}(\mathrm{dist}(i,j) + \mathrm{dist}(j,i)) = B$ and, in turn, that

$$\omega(B) \geq \omega(B^{[j]}(\mathrm{dist}(i,j))) \geq \omega(B).$$

This shows that, if basis i does not change for a duration $2\,\mathrm{diam}(\mathcal{G})+1$, then it will never change afterwards because all bases $B^{[j]}$, for $j \in \{1,\dots,n\}$, have cost equal to $\omega(B)$ at least as early as time equal to $\mathrm{diam}(\mathcal{G})+1$. Therefore, node i can safely stop after a $2\,\mathrm{diam}(\mathcal{G})+1$ duration. ■

7.4 Discussion

We identified a class of distributed optimization problems that appears to be novel and of intrinsic interest. In the next chapter we apply these distributed computation problems in minimum time formation control problems. In particular, we study the rendezvous problem, and the line and circle formation problems. Future directions of research include (i) studying the time complexity of the proposed distributed algorithms, and (ii) finding interesting applications for these optimization problems possibly in the area of sensor networks. We have strong feeling (and some sketches of proof enforcing this feeling) that the time-complexity is $\Theta(n)$.

Chapter 8

Formation control

In this chapter we introduce the problem of minimum time formation for a robotic network. We focus on the formation control problem for a point formation (rendezvous), a line formation and a circle formation. We show that they can be formulated as network abstract linear programs, the exception being the line formation control problem, where suitable conditions on the initial configuration are required. A control and communication law based on the distributed algorithm of the previous chapter is proposed as an approximate solution. This law is basically an extension to this more general setting of the one proposed in [52] for the rendezvous problem.

8.1 Introduction

The main reason for the increasing interest in motion coordination in recent years relies on the desire of trying to accomplish complex tasks by use of many simple and, therefore, cheap systems. An inspiration for this comes from nature where such principle seems to be very common and successful. One of the most interesting and fascinating tasks, very common in nature, is certainly the pattern formation. In particular we ask if formation to simple geometric shapes may be reached by minimizing some objective. In this chapter we are interested in minimum time formation control.

The literature on formation control for robotic networks is vast. Regarding the rendezvous problem, i.e., the problem of gathering the robots at a common location, an early reference is [4]. In this paper Ando and coworkers introduced the "multi-agent rendezvous" problem and a "circumcenter algorithm" to solve it. The algorithm proposed in [4] has been extended to various synchronous and asynchronous stop-and-go strategies in [40]. A related algorithm, in which connectivity constraints are not imposed, is proposed in [41]. In [13] the class of "circumcenter algorithms" has been studied in networks of agents whose state space is $\mathbb{R}^d$, for arbitrary d, and with communication topology characterized by proximity graphs spatially distributed over the disk graph. In [48] the (time and communication) complexity of this and other algorithms has been studied. All these coordination schemes are memoryless (static feedback). In our work we explore dynamic control and communication laws. In particular the laws are based on agreeing on some logic variables and at the same time moving toward the current estimation. A similar approach was used in [54] where the agents try to reach a consensus on a set of variables called coordination variables.

An early reference on distributed algorithms for the formation of geometric patterns is [61]. The "circle formation control" problem, i.e., the problem of steering the robots to a circle formation, is discussed in [15]. A control-Lyapunov function approach to formation control is discussed in [17]. An input-to-state stability approach is taken in [63]. Cyclic formations are studied in [45]. Feasible motions of formations are characterized in [62].

The main contribution of this chapter is the application of the distributed computation problems introduced in the previous chapter in minimum-time formation control problems, such as the rendezvous problem and the problems of line and circle formations. Specifically, we show that the centralized minimum time performance of these three tasks is an abstract linear program. Then we design some joint communication and motion coordinations schemes in which robots move towards the estimated final shape while the final shape is being computed as the solution of a network abstract linear program.

In Section 8.2 we describe the robotic network of first order agents that we use in the formation control scenario. In Section 8.3 and Section 8.4 we introduce the definition of formation task (focusing on rendezvous and on line and circle formation) and we state the minimum time formation problem. Section 8.5 contains the proposed control and communication law. Finally in Section 8.6 we show simulations for the rendezvous case and in Section 8.7 we discuss possible future scenarios.

8.2 Wireless robotic network of first order agents

In this section we briefly describe the model of robotic network that we are using to perform formation control.

The scenario that we want to model is the one of vehicles that move in $\mathbb{R}^d$, $d \in \mathbb{N}$, (usually $d = 2$ or $d = 3$) and communicate by wireless communication. Moreover such vehicles have a bound on the maximum velocity and have the capability of hovering, that is they may hang over in some position with zero velocity.

Remark 8.1 (Examples of vehicles with hovering capability) *Many vehicles have the capability of hovering. For example, mobile wheeled robots have such property. As regards aerial vehicles, helicopters are suitable for this setting, while rigid aircraft are not. Observe that the PVTOL model introduced in Chapter 3 satisfies this condition.* □

One of the simplest deterministic models that captures these properties is a robotic network with agents whose dynamics is described by a first order integrator and that communicate to each other if their distance is less than an upper bound.

Formally, we consider the following network. Each agent i occupies a location $p^{[i]} \in \mathbb{R}^d$, $d \in \mathbb{N}$, and moves according to the first order discrete-time integrator

$$p^{[i]}(t+1) = p^{[i]}(t) + u^{[i]}(t). \tag{8.1}$$

The communication edge map is the one arising according to the *disk graph*, E_{disk}. Each control $u^{[i]}$ takes values in the bounded subset of $\mathbb{R}^d$ $B(0, r_{\text{ctr}})$, that is, $\|u^{[i]}\|_2 \leq r_{\text{ctr}}$. The control and communication law will be defined depending on the coordination task.

We recall that, given $r_{\text{cmm}} \in \mathbb{R}_+$, two agents i and j, $\{i, j\} \subset \{1, \ldots, n\}$, share an edge in the disk graph if and only if

$$\|p_i - p_j\|_2 \leq r_{\text{cmm}} \quad \Longleftrightarrow \quad p_i - p_j \in B(0_d, r_{\text{cmm}}).$$

8.3 Formation tasks

In the literature many definitions of formation have been given and studied. Here we provide a definition of formation based on the network model and the task definition stated in Section 5.4 of Chapter 5. We consider a quite general notion of formation. Roughly speaking, we want to include both the case of agents deployed randomly in a desired subset of the state space and the more rigid scenario of agents fixed in a more structured configuration, e.g., a lattice.

In order to do that we divide the formation task in two tasks. The first is the following. Let $F_0 \subset \mathbb{R}^d$ be a "nominal" subset of the state space and $\alpha \mapsto F_\alpha$, $\alpha \in \mathbb{R}^m$, $m \in \mathbb{N}$ a mapping that provides a parametrization of F_0. For example F_0 could be the x axis in the plane and F_α the set of lines translated and rotated in the plane. We ask the network to reach the configuration where all the agents' states belong to the same subset F_α (for some α). The second task, that could be also missing, regards the configuration of the agents in the subset. In the following we consider only the first task. In order to perform the second one, once the first has been accomplished, one could use for example the deployment control and communication law proposed in [12]. From now on we call formation task only the first part.

Formally, let $\alpha \mapsto F_\alpha$, $\alpha \in \mathbb{R}^m$, $m \in \mathbb{N}$, a parametrization of a nominal set F_0. The *formation task* is defined as

$$\mathcal{T}_{\text{form}}(x) = \begin{cases} \texttt{true}, & \text{if } x^{[i]} \in F_\alpha, x^{[j]} \in F_\alpha \\ & \qquad \text{for some } \ F_\alpha \\ & \qquad \text{for all } \ (i,j) \in E_{\text{cmm}}(x), \\ \texttt{false}, & \text{otherwise.} \end{cases}$$

Remark 8.2 *According to the definition given above, if the graph is not connected, the formation task is achieved if the agents of each connected component belong to the same subset.* □

In the following we are interested in formation to a point (rendezvous), a line and a circle for $d = 2$. Formally, the *rendezvous* task is defined as

$$\mathcal{T}_{\text{rndzvs}}(x) = \begin{cases} \texttt{true}, & \text{if } x^{[i]} = x^{[j]}, \\ & \qquad \text{for all } \ (i,j) \in E_{\text{cmm}}(x), \\ \texttt{false}, & \text{otherwise.} \end{cases}$$

The *line-formation* task is defined as

$$\mathcal{T}_{\text{lform}}(x) = \begin{cases} \texttt{true}, & \text{if } \exists v \in \mathbb{R}^2 \text{ s.t. } \forall \ (i,j) \in E_{\text{cmm}}(x), \\ & \qquad x^{[i]} = x^{[j]} + \alpha v, \quad \alpha \in \mathbb{R} \\ \texttt{false}, & \text{otherwise.} \end{cases}$$

Finally, the *circle-formation* task is defined as

$$\mathcal{T}_{\text{cform}}(x) = \begin{cases} \texttt{true}, & \text{if } \exists p \in \mathbb{R}^2 \text{ s.t. } \forall \ (i,j) \in E_{\text{cmm}}(x), \\ & \quad \|x^{[i]} - p\| = \|x^{[j]} - p\| \\ \texttt{false}, & \text{otherwise.} \end{cases}$$

Remark 8.3 *For* rendezvous, line-formation *and* circle-formation *we could consider the set F_0 as the origin of the plane, the x axis and the unit circle centered at the origin respectively. For the first case the mapping $\alpha \mapsto F_\alpha$ maps the origin into the points of the plane. For the second case it maps the x axis into all the translated and rotated lines in the plane. Finally, for the third case it maps the unit circle into all the translated circles with different radii.* □

8.4 Minimum-time formation

Having defined the formation tasks for the robotic network, we ask whether such tasks can be accomplished in minimum time. The problem may be formalized as follows.

$$\underset{u(\cdot), x(T), T}{\text{minimize}} \quad T,$$

subj. to

(i) $(x(\cdot), u(\cdot))$ is a trajectory of $\mathcal{A} = \{A^{[i]}\}_{i \in I}$;

(ii) i and j can communicate if and only if $(i,j) \in E_{\text{cmm}}(x^{[1]}(t), \dots, x^{[n]}(t))$;

(iii) $\mathcal{T}_{\text{form}}(x(t)) = \texttt{true}$ for all $t \geq T$, $T \in \mathbb{N}$.

We say that a control and communication law $\mathcal{CC}$ is optimal if it solves the above optimal control and communication problem.

An important property of the minimum time formation to a point, a line and a circle is that the centralized version of the problem may be reformulated as an abstract linear program. It turns out that the optimal rendezvous point, the optimal line and the optimal circle are uniquely identified by the smallest enclosing ball, stripe and annulus, respectively, enclosing the n agents. Recall from Chapter 7 that the problems of finding the smallest enclosing ball, smallest enclosing stripe (for points in generic position), and smallest enclosing annulus of a point set are found to be abstract linear programs. The property is formalized in the following proposition. Before stating the proposition, we observe that, for first order integrators, minimum time problems are equivalent to minimum distance ones.

Lemma 8.4 *Let $P \subset \mathbb{R}^d$ be the set of points in $\mathbb{R}^d$, $L \subset \mathbb{R}^2$ the set of lines in $\mathbb{R}^2$ and C the set of circles in $\mathbb{R}^2$. Given the set of points $P_n \subset P$, $P_n = \{p_1, \cdots, p_n\}$, $n \in \mathbb{N}$, consider the problems:*

(i) $\min\limits_{p \in P} \max\limits_{p_j \in P_n} \|p_j - p\|$;

(ii) $\min\limits_{l \in L} \max\limits_{p_j \in P_n} \operatorname{dist}(p_j, l)$, $d = 2$;

(iii) $\min\limits_{c \in C} \max\limits_{p_j \in P_n} \operatorname{dist}(p_j, c)$, $d = 2$;

where, given a set S and a point p, $\operatorname{dist}(p, S)$ is the distance of p from S, that is $\operatorname{dist}(p, S) = \min\limits_{s \in S} \|p - s\|$.

These are equivalent to the problems of finding the smallest enclosing ball, stripe and annulus of the point set, which are abstract linear programs. For Problem (ii) we need the further assumption that the point set is in generic position. □

Remark 8.5 (Generic positions and locality assumption) *As regards the smallest enclosing stripe problem, the assumption that the points are in generic positions ensures that, for any subset of points, the smallest enclosing stripe is unique, thus saving the locality property that is not true in the general setting. As counter example, it suffices taking the points on the vertices of a regular polygon.* □

8.5 Move-toward-estimate control and communication law

We have shown that the centralized solution of the minimum time formation to point, line and circle can be found by solving an abstract linear program. In Chapter 7 we have shown that given a network and an abstract linear program we can define a network abstract linear program which is basically a distributed version of the optimization problem over the network. We have also provided a distributed algorithm (with two possible variants) to solve it. With such tool in hand we can build a control and communication law that approximates the optimal solution of the minimum time formation problem by "emulating" the centralized solution. The law is based on the scheme depicted in Figure 8.1. In the task layer, that coincides with the logical layer

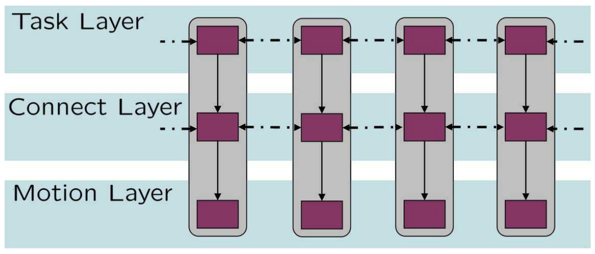


Figure 8.1: Hierarchical scheme of the control and communication law

(recall Section 5.4 in Chapter 5), each agent runs the *FloodBasis* algorithm to solve the network abstract linear program. At each communication round the current estimate of the solution is provided to the physical layer (connectivity and motion layers). Using this estimate each agent computes, at each instant, a target state. This target state is exactly the final state to reach if the execution of the *FloodBasis* is over. This signal is filtered by the connectivity layer. Each agent modifies the target state so that network connectivity is maintained. Finally the motion layer tracks the "filtered" target point computed by the connectivity layer. In order to speed up the process and to guarantee convergence, the connectivity layer is bypassed once the task layer reaches the halting condition (meaning that the network abstract linear program has been solved).

Before stating the algorithm formally, we need to describe how connectivity is maintained in first order networks. The idea has been already introduced in Section 6.2 of Chapter 6.

In a network with communication edge map $E_{\text{cmm}} = E_{\text{disk}}$, if agents i and j are neighbors at time $t \in \mathbb{N}_0$, then we require their subsequent positions to belong to $B(\frac{p^{[i]}(t)+p^{[j]}(t)}{2}, \frac{1}{2}r_{\text{cmm}})$. If an agent i has its neighbors at locations

$\{q_1, \dots, q_l\}$ at time t, then its *constraint set* $\mathcal{D}_{x^{[i]}(t), r_{\text{cmm}}}(\{q_1, \dots, q_l\})$ is

$$\mathcal{D}_{p^{[i]}(t),r}(\{q_1, \dots, q_l\}) = \bigcap_{q \in \{q_1, \dots, q_l\}} B\Big(\frac{p^{[i]}(t) + q}{2}, \frac{1}{2} r_{\text{cmm}}\Big).$$

In order to maximize the displacement toward the desired point, each agent solves a convex optimization problem that can be stated in general as follows. For q_0 and q_1 in $\mathbb{R}^d$, and for a convex closed set $Q \subset \mathbb{R}^d$ with $q_0 \in Q$, let $\lambda(q_0, q_1, Q)$ denote the solution to the strictly convex problem:

$$\begin{aligned} \text{maximize } & \lambda \\ \text{subj. to } & \lambda \leq 1, \ (1 - \lambda) q_0 + \lambda q_1 \in Q. \end{aligned}$$

We call $\lambda_*(x^{[i]}, x_{\text{trgt}})$ the solution of the convex problem for $q_0 = x^{[i]}$, $q_1 = x_{\text{trgt}}$ and $Q = \mathcal{D}_{x^{[i]}(t), r_{\text{cmm}}}\big(\{p^{[j]}\}_{j \in \mathcal{N}(i)}\big)$.

One more notion is needed. Let $\mathcal{T}_{\text{form}}$ be either $\mathcal{T}_{\text{rndzvs}}$, $\mathcal{T}_{\text{lform}}$ or $\mathcal{T}_{\text{cform}}$, and B the basis that solves the associated network abstract linear program. The function `target`(B) computes the center of the minimal enclosing ball, the central line of the minimal enclosing stripe and the central circle of the minimal enclosing annulus respectively.

The control and communication law is described formally in the following table.

Remark 8.6 *The move-toward-estimate control and communication law, as stated in the table, does not guarantee connectivity of the network once formation has been reached. For the rendezvous case connectivity is trivially obtained (the agents are at the same point). For the line-formation it can be easily shown that the same holds. In fact, once the halting condition has been reached, the agents are connected. But, using a reference system with the horizontal axis coincident with the optimal line, it turns out that the final distance between the agents (when they reach the optimal line) is just the projection of the distance, at the halting instant, on the horizontal axis. Finally, for the circle, connectivity is not guaranteed in general, but it can be easily regained by use of a deployment algorithm (the agents are on a bounded set).* □

The correctness of the control and communication laws is proven in the following proposition.

Proposition 8.7 (Move-toward-estimate correctness) *On the network $\mathcal{S}$ with communication edge map E_{disk} and bound on the ith control input*

`Control and comm. law:`	*Move-toward-estimate*
`Goal:`	Approximate minimum time formation
`Message alphabet:`	$M = H \cup \{\texttt{null}\},\ \ H = \{x^{[i]}(0)\}_{i\in\{1,\dots,n\}}$
`Logical state:`	$w^{[i]} = (B^{[i]}, h^{[i]}, \texttt{halt}^{[i]}),\ \ B^{[i]} \in H^{\delta},\ h^{[i]} \in H$
`Physical state:`	$x^{[i]} = p^{[i]},\ \ p^{[i]} \in \mathbb{R}^d$
`Initialization:`	$B^{[i]} = (x^{[i]}(0), \cdots, x^{[i]}(0)),\ h^{[i]} = x^{[i]}(0),$ $\texttt{halt}^{[i]} = 0$

```
function msg($w^{[i]}, j$)
 1: if $\texttt{halt}^{[i]} < 2\,\mathrm{diam}(\mathcal{G}(0))$ then
 2:     return $B^{[i]}$
 3: else
 4:     return null
 5: end if
function stf($w^{[i]}, y$)
 1: collect $y^{[i]} := \{\mathrm{msg}(w^{[j]}, i) \mid j \in \mathcal{N}(i)\}$
 2: collect $H_i := (y^{[i]}, w^{[i]})$
 3: save $B^{[i]}_{\mathrm{temp}} := B^{[i]}$
 4: compute $B^{[i]} := \texttt{SUBEX\_lp}(H_i, B^{[i]})$
 5: if $B^{[i]} = B^{[i]}_{\mathrm{temp}}$ & $\texttt{halt}^{[i]} < 2\,\mathrm{diam}(\mathcal{G}(0))$ then
 6:     $\texttt{halt}^{[i]} = \texttt{halt}^{[i]} + 1$
 7: else if $\texttt{halt}^{[i]} < 2\,\mathrm{diam}(\mathcal{G}(0))$ then
 8:     $\texttt{halt}^{[i]} = 0$
 9: else
10:     $\texttt{halt}^{[i]} = \texttt{halt}^{[i]}$
11: end if
12: return $(B^{[i]}, h^{[i]}, \texttt{halt}^{[i]})$
function ctl($x^{[i]}, w^{[i]}, y$)
 1: $S_{\texttt{trgt}} = \texttt{target}(B^{[i]})$
 2: $p_{\mathrm{trgt}} = \arg\min_{p\in S_{\mathrm{trgt}}} \|x^{[i]} - p\|$
 3: if $\texttt{halt}^{[i]} < 2\,\mathrm{diam}(\mathcal{G}(0))$ then
 4:     $\lambda = \lambda_*(x^{[i]}, p_{\mathrm{trgt}})$
 5: else
 6:     $\lambda = 1$
 7: end if
 8: return $\max\{\lambda \cdot (p_{\mathrm{trgt}} - x^{[i]}), r_{\mathrm{ctr}}\} \cdot \mathrm{vers}(p_{\mathrm{trgt}} - x^{[i]})$
```

$u^{[i]} \in B(0, r_{\mathrm{ctr}})$, the move-toward-estimate control and communication laws achieve the task $\mathcal{T}_{\mathrm{rndzvs}}$, $\mathcal{T}_{\mathrm{lform}}$ and $\mathcal{T}_{\mathrm{cform}}$ respectively. For the line-formation task we need the further assumption that the point set of initial conditions is in generic position. □

Proof: By the connectivity arguments done before and by Proposition 7.6 we know that there exists $\overline{T} \in \mathbb{N}$ such that for $t = \overline{T}$ the network is connected and all the agents have solved the network abstract linear program. Since this instant all the agents can move toward the target set (point, line or circle) at maximum speed without enforcing connectivity constraint anymore. Thus, they achieve the task. ■

8.6 Simulations

In this section we show simulations for the rendezvous problem. We implemented the move-toward-estimate law, based on the *FloodBasis* algorithm for the smallest enclosing ball problem (we call it *FloodMEB*). We implemented it in the plane, $d = 2$, over the disk graph. In [52] a detailed version of the algorithm can be found together with its equivalent version for the problem with bounds on the infinity norm of the control.

The simulation run is illustrated in Figure 8.2. The 32 agents have a bound on the control inputs $r_{\mathrm{ctr}} = 0.1$, and a communication radius $r_{\mathrm{cmm}} = 3$. The initial positions of the agents were randomly generated over the rectangle $[-6, 6] \times [-3, 3]$.

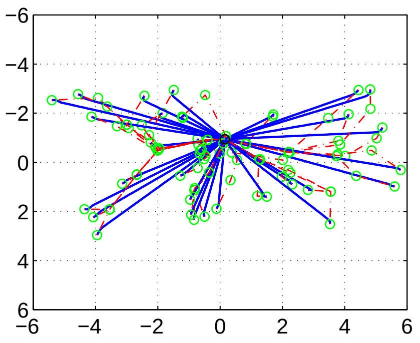


Figure 8.2: Evolution of the network (in filled blue) with evolution of *FloodMEB* (green circles connected by dashed red line)

The *FloodMEB* law converges in five steps, while the rendezvous is achieved at $T = 58$. As it clearly appears in the figure, once the consensus on the minimal enclosing ball is reached, all the agents move toward the center.

8.7 Discussion

We provided a distributed control and communication law to perform rendezvous and, line and circle formation. The law is based on the distributed algorithms introduced in the previous chapter to solve network abstract linear programming. Future directions of this work include: (i) applying the proposed law to networks whose agents have more complex dynamics, e.g. a second order integrator, (iii) studying the complexity of this law and looking for algorithms that solve the exact minimum time problem, (ii) use the distributed algorithm for network abstract linear programming to estimate a shape that includes the agents at each time-instant (e.g. the smallest ball).

Perspectives

In the first part of the dissertation we studied the problem of exploring feasible trajectories of nonlinear control systems, that is trajectories satisfying state and input constraints. We developed an effective strategy that was successfully applied to the simplified PVTOL aircraft model. An important direction of investigation is in the area of Receding Horizon Control. It includes (i) developing a receding horizon scheme, based on the same optimization technique as the exploration strategy, that allows to track feasible trajectories while satisfying the state and input constraint, (ii) designing a hierarchical strategy that implements both the exploration and the tracking tasks and proving its effectiveness and correctness. Another direction of research regards the applications of such strategy. It is under development a simulation test bed for trajectory exploration and tracking of a rigid aircraft model.

We have also introduced the novel notion of *operating region* meant as a region where trajectories are ensured to be exponentially stabilizable. The characterization of this region for control affine systems is still preliminary. We aim to provide sufficient conditions to characterize an operating region for control affine systems driven by (essentially) bounded inputs.

In the second part of the thesis we studied optimization problems in robotic networks. First, we provided distributed algorithms to enforce connectivity among networks of agents with double-integrator dynamics. Future directions of research include (i) evaluating the communication complexity of the proposed distributed algorithm and (ii) investigating the flocking phenomenon emerging from the enforcement of connectivity. Second, we identified a class of distributed optimization problems that appears to be novel and of intrinsic interest. We have applied these distributed computation problems in minimum time formation control problems. Future directions of research include (i) studying the time complexity of the proposed distributed algorithms, and (ii) finding interesting applications for these optimization problems possibly in the area of sensor networks.

Appendix A

Projected Jacobi method

We briefly review here a parallel algorithm for the solution of a quadratic optimization problem. The technique is known as the *projected Jacobi method* in the literature on network flow control problems ([6], Section 3.4).

Consider the quadratic programming problem

$$\begin{aligned} \text{minimize } & \frac{1}{2}x^TQx - b^Tx, \\ \text{subj. to } & Ax \preceq c, \end{aligned}$$

where Q is a given $n \times n$ symmetric positive definite matrix, A is a given $m \times n$ matrix, and $b \in \mathbb{R}^n$ and $c \in \mathbb{R}^m$ are given vectors. The dual problem is

$$\begin{aligned} \text{minimize } & \frac{1}{2}y^TFy + s^Ty, \\ \text{subj. to } & y \succeq 0, \end{aligned}$$

for $F = AQ^{-1}A^T$ and $s = c - AQ^{-1}b$. If y^* solves the dual problem, then $x^* = Q^{-1}(b - A^Ty^*)$ solves the primal problem.

For a step size parameter $\tau > 0$ and for $j \in \{1, \ldots, n\}$, the projected Jacobi iteration, when the j^{th} coordinate is updated, has the form

$$y_j(t+1) = \max\Big\{0, y_j(t) - \frac{\tau}{f_{jj}}\Big(s_j + \sum_{k=1}^{m} f_{jk}y_k(t)\Big)\Big\}, \tag{A.1}$$

where f_{jk} is the j, k^{th} element of the matrix F. As discussed in [6], this algorithm converges to the global solution of the dual problem if the step size τ is chosen sufficiently small; in particular, convergence is guaranteed for $\tau = 1/m$.

Appendix B

Appendix on Shostak's test

This section provides a proof for Theorem 6.12. The proof amounts to showing that if E is the edge set of a spanning tree T in $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k) r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i \in \{1,\dots,n\}}$, then the control constraint set $\mathcal{U}_E^d(r_{\text{cmm}}, r_{\text{ctr}}, \nu(k)) \cdot (\{p^{[i]}, v^{[i]}\}_{i \in \{1,\dots,n\}})$ is non-empty. We first consider a polytopic approximation of constraints (6.10) and (6.11). Among all possible choices, we use the conservative orthotope approximation that allows us to decouple the constraints into d independent sets of linear inequalities (one for each dimension). Then we use Shostak's theory to obtain sufficient conditions for the feasibility of these linear inequalities. For brevity, we drop the dependence of the quantities on t and we assume that the variables $u^{[i]}$ are scalars, for all $i \in \{1, \dots, n\}$ and $t \geq 0$. The resulting sets of linear inequalities for one particular dimension are

$$\delta_{i,j}^l \leq u^{[i]} - u^{[j]} \leq \delta_{i,j}^u, \quad \text{and} \quad -\frac{r_{\text{ctr}}}{\sqrt{d}} \leq u^{[i]} \leq \frac{r_{\text{ctr}}}{\sqrt{d}}. \tag{B.1}$$

where $-\nu(k) r_{\text{ctr}} \leq \delta_{i,j}^l \leq \delta_{i,j}^u \leq \nu(k) r_{\text{ctr}}$, for all $i, j \in \{1, \dots, n\}$ and $i \neq j$.

B.0.1 Shostak Theory

In this section we present Shostak's theory for feasibility of linear inequalities involving at most two variables, similar to the ones in (B.1). These ideas will then be used to prove Theorem 6.12. The notations used in [5] adapted to our case are presented next. Let $u^{[0]}$ be an auxiliary *zero variable* that always occurs with zero coefficient - the only variable that can do this. Without loss of generality, we can thus assume that all the inequalities in $\mathcal{L}$ contain two variables. As a result of this, the inequalities in (B.1) can be succinctly written as

$$u^{[i]} - u^{[j]} \leq \delta_{i,j}, \quad \forall i, j \in \{0, \dots, n\}, \tag{B.2}$$

where for all $i, j \in \{1, \dots, n\}$, $i \neq j, -\nu(k)r_{\text{ctr}} \leq \delta_{i,j} \leq \nu(k)r_{\text{ctr}}$ and for all $i \in \{1, \dots, n\}, \delta_{i,0} = \delta_{0,i} = \frac{r_{\text{ctr}}}{\sqrt{d}}$. Also implicit in this formulation is the relation that $\delta_{i,j} + \delta_{j,i} \geq 0$ for all $i, j \in \{0, \dots, n\}$ and $i \neq j$.

Let $\mathcal{L}$ denote the system of inequalities in (B.2). We construct the graph $G(\mathcal{L})$ with $n+1$ vertices and $2(2n-1)$ edges as follows: (a) For each variable $u^{[i]}$ occurring in $\mathcal{L}$, add a vertex i to $G(\mathcal{L})$. (b) For each inequality of the form $a_{i,j}u^{[i]} + b_{i,j}u^{[j]} \leq \delta_{i,j}$ in $\mathcal{L}$, add an undirected edge between i and j to $G(\mathcal{L})$, and label the edge with the inequality (see Figure B.1). It is easy to see the following relation between the spanning tree T in $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k)r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i \in \{1,\dots,n\}}$ that is used to derive the constraints in the inequalities (B.2) and the graph $G(\mathcal{L})$: (a) The vertex set of $G(\mathcal{L})$ is the union of the vertex set of T and the auxiliary vertex 0 (b) For every edge $\{i, j\}$ in T, there are two edges between the vertices i and j in $G(\mathcal{L})$ (c) Additionally, $G(\mathcal{L})$ contains two edges between 0 and every other vertex i, for all $i \in \{1, \dots, n\}$.

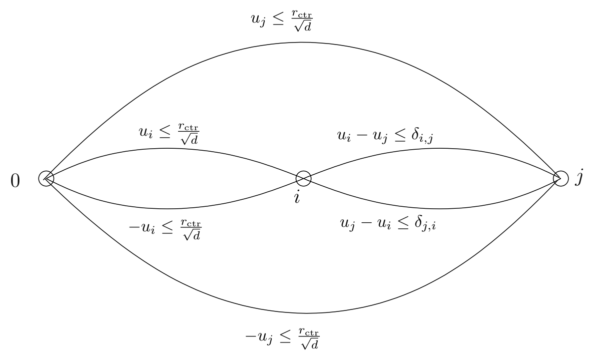

$u_j \leq \frac{r_{\text{ctr}}}{\sqrt{d}}$
$u_i \leq \frac{r_{\text{ctr}}}{\sqrt{d}}$
$u_i - u_j \leq \delta_{i,j}$
0
i
j
$-u_i \leq \frac{r_{\text{ctr}}}{\sqrt{d}}$
$u_j - u_i \leq \delta_{j,i}$
$-u_j \leq \frac{r_{\text{ctr}}}{\sqrt{d}}$

Figure B.1: Snippet of the graph $G(\mathcal{L})$ for the system of inequalities in (B.2)

To every edge represented by the inequality of the form $a_{i,j}u^{[i]} + b_{i,j}u^{[j]} \leq \delta_{i,j}$, we associate *a triple* $\langle a_{i,j}, b_{i,j}, \delta_{i,j} \rangle$. Note that $\langle b_{i,j}, a_{i,j}, \delta_{i,j} \rangle$ is also a triple associated with the same edge. Without loss of generality, consider a path of $G(\mathcal{L})$ determined by the vertices $\{1, 2, \dots, l+1\}$ and the edges $e_{1,2}, e_{2,3}, \dots, e_{l,l+1}$ between them. *A triple sequence*, P, associated with the path is defined as

$$\langle a_{1,2}, b_{1,2}, \delta_{1,2} \rangle, \langle a_{2,3}, b_{2,3}, \delta_{2,3} \rangle, \dots, \langle a_{l,l+1}, b_{l,l+1}, \delta_{l,l+1} \rangle,$$

where, for $1 \leq i \leq l$, $a_{i,i+1}u^{[i]} + b_{i,i+1}u^{[j]} \leq \delta_{i,i+1}$ is the inequality associated with the edge $e_{i,i+1}$. If $a_{i+1,i+2}$ and $b_{i,i+1}$ have opposite signs for $1 \leq i < l$, then P is called *admissible*.

Define $\langle a_P, b_P, \delta_P \rangle$, the *residue* of P, as

$$\langle a_P, b_P, \delta_P \rangle = \langle a_{1,2}, b_{1,2}, \delta_{1,2} \rangle \odot \langle a_{2,3}, b_{2,3}, \delta_{2,3} \rangle \odot \ldots \odot \langle a_{l,l+1}, b_{l,l+1}, \delta_{l,l+1} \rangle,$$

where $\odot$ is the associativity binary operator defined on triples by

$$\langle a, b, \delta \rangle \odot \langle a', b', \delta' \rangle = \langle \kappa a a', -\kappa b b', \kappa(\delta a' - \delta' b) \rangle, \qquad \text{where} \quad \kappa = a'/|a'|.$$

Intuitively, the operator $\odot$ takes two inequalities and derives a new inequality by eliminating a common variable; e.g., $ax+by \leq \delta$ and $a'y+b'z \leq \delta'$ imply $-aa'x + bb'z \leq -(\delta a' - \delta' b)$ if $a < 0$ and $b > 0$. Note that the signs of a_P and $a_{1,2}$ agree, as do the signs of b_P and $b_{1,2}$.

A path is called a *loop* if the initial and final vertices are identical. (A loop is not uniquely specified unless its initial vertex is given.) If all the intermediate vertices of a path are distinct, the path is *simple.* An admissible triple sequence P associated with a loop with initial vertex x is *infeasible* if its residue satisfies $a_P + b_P = 0$ and $\delta_P < 0$. A loop which contains an infeasible triple sequence is called an *infeasible loop.* Thus if $G(\mathcal{L})$ has an infeasible loop, the system of inequalities $\mathcal{L}$ is unsatisfiable. However, the converse is not true in general. Next, we show how to extend $\mathcal{L}$ to an equivalent system $\mathcal{L}'$ such that $G(\mathcal{L}')$ has an infeasible simple loop if and only if $\mathcal{L}$ is unsatisfiable.

For each vertex i of $G(\mathcal{L})$ and for each admissible triple sequence P with $a_P + b_P \neq 0$ associated with a simple loop of $G(\mathcal{L})$ and initial vertex i, add a new inequality $(a_P + b_P)u^{[i]} \leq \delta_P$ to $\mathcal{L}$. This new system $\mathcal{L}'$ is referred to as the *Shostak extension* of $\mathcal{L}$. We now state the necessary and sufficient condition on the extended system of inequalities $\mathcal{L}'$ for the satisfiability of the original system $\mathcal{L}$.

Theorem B.1 (Shostak's Theorem [5]) *Let $\mathcal{L}'$ be the Shostak extension of $\mathcal{L}$. The system of inequalities $\mathcal{L}$ is satisfiable if and only if $G(\mathcal{L}')$ contains no infeasible simple loop.*

B.0.2 Satisfiability test

In this section we use the Shostak criterion to derive conditions for the satisfiability of the inequalities in (B.2).

Lemma B.2 *Let $\mathcal{L}$ be the system of inequalities of the form* (B.2) *obtained by considering pairwise neighbors in a spanning tree T in $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k)r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$. Then the Shostak extension of $\mathcal{L}$ is itself.*

Proof: Consider a simple loop of $G(\mathcal{L})$ with the initial vertex $i \in \{0, 1, \dots, n\}$. Consider an admissible triple sequence P associated with the loop. Since $a_{i,j}, b_{i,j} \in \{-1, +1\}$, for all $i, j \in \{1, \dots, n\}, i \neq j$, and $a_{0,i}, a_{i,0}, b_{i,0}, b_{0,i} \in \{-1, 0, +1\}$, for all $i \in \{1, \dots, n\}$, the residue of P, $\langle a_P, b_P, \delta_P\rangle$, is such that $a_p + b_p = 0$. Hence, no new inequality must be added to obtain the Shostak extension of $\mathcal{L}$. ■

Lemma B.3 *Let $\mathcal{L}$ be the system of inequalities of the form* (B.2) *obtained by considering pairwise neighbors in a spanning tree T of depth at most k in $\mathcal{G}_{\text{di-disk}}(r_{\text{cmm}}, \nu(k)r_{\text{ctr}})$ at $\{(p^{[i]}, v^{[i]})\}_{i\in\{1,\dots,n\}}$. If $\nu(k) = \frac{2}{k\sqrt{d}}$, then there is no infeasible simple loop in $G(\mathcal{L})$.*

Proof: Looking at figure B.1 it is clear that there are two types of simple loops with admissible triple sequences in $G(\mathcal{L})$:

(i) $\langle +1, -1, \delta_{i,j}\rangle$, $\langle +1, -1, \delta_{j,i}\rangle$ or $\langle -1, +1, \delta_{i,j}\rangle$, $\langle -1, +1, \delta_{j,i}\rangle$, where $i, j \in \{0, \dots, n-1\}$ and $\{i, j\}$ is an edge in T.

(ii) $\langle 0, -1, \frac{r_{\text{ctr}}}{\sqrt{d}}\rangle, \langle +1, -1, \delta_{i_1,i_2}\rangle, \dots, \langle +1, -1, \delta_{i_{l-1},i_l}\rangle, \langle +1, 0, \frac{r_{\text{ctr}}}{\sqrt{d}}\rangle$ or $\langle 0, +1, \frac{r_{\text{ctr}}}{\sqrt{d}}\rangle, \langle -1, +1, \delta_{i_2,i_1}\rangle, \dots, \langle -1, +1, \delta_{i_l,i_{l,l-1}}\rangle, \langle -1, 0, \frac{r_{\text{ctr}}}{\sqrt{d}}\rangle$, where $i_l \in \{1, \dots, \zeta\}$ for all $l \in \{1, \dots, \zeta\}$ and $\{i_l, i_{l+1}\}$ is an edge in T.

The residue for the first set of loops is $\langle +1, -1, \delta_{i,j} + \delta_{j,i}\rangle$ or $\langle -1, +1, \delta_{i,j} + \delta_{j,i}\rangle$. The feasibility condition is trivially satisfied by construction since $\delta_{i,j} + \delta_{j,i} \geq 0$. For the second set of loops, the residue is:

$$\Big\langle 0, -1, \frac{r_{\text{ctr}}}{\sqrt{d}}\Big\rangle \odot \langle +1, -1, \delta_{i_1,i_2}\rangle \odot \dots \odot \langle +1, -1, \delta_{i_{\zeta-1},i_\zeta}\rangle \odot \Big\langle +1, 0, \frac{r_{\text{ctr}}}{\sqrt{d}}\Big\rangle$$
$$= \Big\langle 0, 0, 2\frac{r_{\text{ctr}}}{\sqrt{d}} + \sum_{l=1}^{\zeta-1} \delta_{i_l,i_{l+1}}\Big\rangle,$$

or

$$\Big\langle 0, +1, \frac{r_{\text{ctr}}}{\sqrt{d}}\Big\rangle \odot \langle -1, +1, \delta_{i_2,i_1}\rangle \odot \dots \odot \langle -1, +1, \delta_{i_\zeta,i_{\zeta-1}}\rangle \odot \Big\langle -1, 0, \frac{r_{\text{ctr}}}{\sqrt{d}}\Big\rangle$$
$$= \Big\langle 0, 0, 2\frac{r_{\text{ctr}}}{\sqrt{d}} + \sum_{l=1}^{\zeta-1} \delta_{i_l,i_{l+1}}\Big\rangle.$$

In order to guarantee the feasibility of the second set of loops, we need that $2\frac{r_{\mathrm{ctr}}}{\sqrt{d}} + \sum_{l=1}^{\zeta-1} \delta_{i_l,i_{l+1}} \geq 0$. We derive conditions for the worst case which occurs when the loop is written for the longest path in T, i.e., when $\zeta = k+1$ and when $\delta_{i_l,i_{l+1}} = -\nu(k)r_{\mathrm{ctr}}$, for all $l \in \{1, \ldots, k\}$. In this case, there is no infeasible simple loop if and only if

$$2\frac{r_{\mathrm{ctr}}}{\sqrt{d}} - k\nu(k)r_{\mathrm{ctr}} \geq 0,$$

that is, if and only if $\nu(k) = \frac{2}{k\sqrt{d}}$. ■

Finally, the proof of Theorem 6.12 follows from Theorem B.1, Lemma B.2 and Lemma B.3.

Acknowledgements

My first thanks go to my advisor, Professor Frezza, for giving me the great opportunity of working with him. Thanks for believing in me since the beginning, when even I would not have bet on myself. Thanks for supporting me in any occasion and for giving me the unique opportunity of visiting new exciting places, and collaborating with or just meeting some of the best professors in the world. Thanks for the numerous interesting discussions and for sharing with me, in front of a good coffee, his farseeing and always successful ideas. I can say without any doubt that I owe him any goal I reached in these years.

I want to thank both my co-advisors, Professor John Hauser and Professor Francesco Bullo, for accepting this crazy idea of working "by Skype" (and thanks to Skype for existing).

Thanks to John, a genius in my opinion (and sometimes even a genie for me). Thanks for introducing me to the wonders of nonlinear optimization. Thanks for sharing with me his great ideas and his powerful tools (his creatures!). I also want to thank him for being such a wonderful host. Now, I understand what Ale meant by "making me feel at home in Colorado". I will never forget the nights spent discussing about beautiful math, after a delicious barbecue (with fantastic asparagus) and with a glass of excellent wine in hand. Thanks for teaching me that without a great intuition you cannot have great conjectures, but that if you do not follow a rigorous mathematical argument they will never become theorems. Thanks for helping me with many proofs and especially for working until the last day, also during the night, to allow me to have an elegant version of the proof for the main theorem of Chapter 2.

Thanks to Francesco, for accepting me as visiting student at UCSB, thus giving me the opportunity of living one of the best experiences in my life. Thanks for considering me as one of his students. Thanks for teaching me rigor and methodology, essential features to reach important results, especially working with math. Thanks for reminding me that quality is much more important than quantity and that a good story needs also to be told in an elegant manner. Thanks for driving me in the interesting topics of motion coordination, for suggesting me the right directions of research and for helping me in proving the needed results with clean arguments.

Remaining in Santa Barbara, a special thank goes to Daniele, Dmitry, Max and Sarah for being my family in those six months that I spent there. Thanks to Carlo for coming to the other side of the world to visit me. I know friendship is not measured in mileage, but I think this gives a good approximation. Thanks all you guys for those beautiful and amazing moments that we lived together, for sure some of the best in my life. I feel happy just remembering those moments. I am sure they will remain fixed in my memory forever.

Thanks to my lab-mates at UCSB, Chunkai, Anurag, Ketan, Sonia and Sara. In particular, thanks to Ketan for working with me. Chapter 6 is the result of our joint work supervised by Francesco.

Back to Padova, I want to thank Professor Picci, for being the first one to welcome me in this wonderful research group. Thanks to Ale Beghi for his cheeriness. Sometimes he can let you smile even in a bad day. And then thanks to all the other professors of the group. I have a special thank to Professor Marchesini for reminding me, by his magistery, that I had to deserve to be part of this excellent group.

Thanks to Ale, Gege and Checco for the numerous nights spent waiting for the "sheriff" reminding us, with the sound of the keys, that we had to go home. Thanks to Nicuz, Ticoz and Simone for starting with me this wonderful adventure. And thanks to Andrea, Stefano, Riccardo, Marione, Alex, Maura, Paolo, Rugio, Stefano, Martina, Federico, Moretto, Angelo, Alberto, Simone (hope not to forget anybody) and all the other persons that contributed to have a really friendly atmosphere in the lab during these three years.

Back once more, to Pisa, I want to thank Professor Bicchi for his regard for me and for transmitting me the passion for nonlinear control. If I reached this point, in part, I owe it to him as well. And thanks to Lucia for reminding me (three years ago) that sometimes just going upstairs rather than downstairs can completely change your life. She was definitely right.

Thanks to my parents and to my brother for their support in these years.

Finally, the most important thanks are for Mariella. First, because without her, almost surely, I would not have come to Padova and met most of the persons I have thanked. But, much more, thanks for being with me in any important moment and for being happy for me even when I had to leave her for six months or the week before her graduation (Ok, maybe in that occasion you were not so happy!). Thank you so much!

Bibliography

[1] P. K. Agarwal and S. Sen. Randomized algorithms for geometric optimization problems. In P. Pardalos, S. Rajasekaran, J. Reif, and J. Rolim, editors, *Handbook of Randomization*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[2] Miklos Ajtai and Nimrod Megiddo. A deterministic poly$(\log\log n)$-time n-processor algorithm for linear programming in fixed dimension. *SIAM Journal on Computing*, 25(6):1171–1195, 1996.

[3] S. A. Al-Hiddabi and N. H. McClamroch. Tracking and maneuver regulation control for nonlinear nonminimum phase systems: application to flight control. *IEEE Transactions on Control Systems Technology*, 10(6), 2002.

[4] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Transactions on Robotics and Automation*, 15(5):818–828, 1999.

[5] B. Aspvall and Y. Shiloach. A polynomial time algorithm for solving systems of linear inequalities with two variables per inequality. *SIAM Journal on Computing*, 9(4):827–845, 1980.

[6] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Athena Scientific, Belmont, MA, 1997.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, Cambridge, UK, 2004.

[8] M. Bui, F. Butelle, and C. Lavault. A distributed algorithm for constructing a minimum diameter spanning tree. *Journal of Parallel and Distributed Computing*, 64:571–577, 2004.

[9] G. Buttazzo, M. Giaquinta, and S. Hildebrandt. *One-dimensional Variational Problems.* Oxford University Press, New York, 1998.

[10] L. Cesari. *Optimization - Theory and Applications: Problems with Ordinary Differential Equations.* Springer-Verlag, New York, 1983.

[11] J. M. Coron. Linearized control systems and applications to smooth stabilization. *SIAM Journal on Control and Optimization*, 32(2):358–386, 1994.

[12] J. Cortés, S. Martínez, and F. Bullo. Spatially-distributed coverage optimization and control with limited-range interactions. *ESAIM. Control, Optimisation & Calculus of Variations*, 11:691–719, 2005.

[13] J. Cortés, S. Martínez, and F. Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automatic Control*, 51(8):1289–1298, 2006.

[14] G. De Nicolao, L. Magni, and R. Scattolini. Stabilizing receding-horizon control of nonlinear time-varying systems. *IEEE Transactions on Automatic Control*, 43(7):1030–1036, 1998.

[15] X. Defago and A. Konagaya. Circle formation for oblivious anonymous mobile robots with no common sense of orientation. In *ACM International Workshop on Principles of Mobile Computing (POMC 02)*, pages 97–104, Toulouse, France, October 2002.

[16] S. Devasia, D. Chen, and B. Paden. Nonlinear inversion-based output tracking. *IEEE Transactions on Automatic Control*, 41(7), 1996.

[17] M. Egerstedt and X. Hu. Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, 17(6):947–951, 2001.

[18] R. A. Freeman and P. V. Kokotovic. *Robust Nonlinear Control Design State-Space and Lyapunov Techniques.* Birkhäuser, Boston, 1996.

[19] A. Ganguli, J. Cortés, and F. Bullo. On rendezvous for visually-guided agents in a nonconvex polygon. In *IEEE Conf. on Decision and Control and European Control Conference*, pages 5686–5691, Seville, Spain, December 2005.

[20] Z. Gao. On discrete time optimal control: A closed-form solution. In *American Control Conference*, pages 52–58, Boston, MA, 2004.

[21] Bernd Gärtner. A subexponential algorithm for abstract optimization problems. *SIAM Journal on Computing*, 24(5):1018–1035, 1995.

[22] Bernd Gärtner and Emo Welzl. Linear programming - randomization and abstract frameworks. In *Symposium on Theoretical Aspects of Computer Science*, volume 1046 of *Lecture Notes in Computer Science*, pages 669–687, 1996.

[23] N. H. Getz and J. E. Marsden. Control for an autonomous bicycle. In *IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 1397–1402, May 1995.

[24] G. Grimm, M. Messina, A. R. Teel, and S. Tuna. Model predictive conrol: for want of a control Lyapunov function, all is not lost. *IEEE Transactions on Automatic Control*, 50:546–558, 2005.

[25] J. Hauser. A projection operator approach to the optimization of trajectory functionals. In *IFAC World Congress*, Barcellona, 2002.

[26] J. Hauser and R. Hindman. Maneuver regulation from trajectory tracking: Feedback linearizable systems. In *IFAC Symposium on Nonlinear Control Systems*, pages 638–643, Tahoe City, CA, June 1995.

[27] J. Hauser and D. G. Meyer. The trajectory manifold of a nonlinear control system. In *IEEE Conf. on Decision and Control*, volume 1, pages 1034–1039, December 1998.

[28] J. Hauser and D. G. Meyer. Trajectory morphing for nonlinear systems. In *American Control Conference*, 1998.

[29] J. Hauser and A. Saccon. A barrier function method for the optimization of trajectory functionals with constraints. In *IEEE Conf. on Decision and Control*, pages 864–869, San Diego, Dec 2006.

[30] J. Hauser, A. Saccon, and R. Frezza. Aggressive motorcycle trajectories. In *IFAC Symposium on Nonlinear Control Systems*, Stuttgart, 2004.

[31] J. Hauser, A. Saccon, and R. Frezza. On the driven inverted pendulum. In *IEEE Conf. on Decision and Control*, pages 6176–6180, Dec. 2005.

[32] J. Hauser, S. S. Sastry, and G. Meyer. Nonlinear control design for slightly nonminimum phase systems: Application to V/STOL aircraft. *Automatica*, 28(4):665–679, 1992.

[33] R. Hermann and A. J. Krener. Nonlinear controllability and observability. *IEEE Transactions on Automatic Control*, 22:728–740, 1977.

[34] H. Hermes. On local and global controllability. *SIAM Journal on Control*, 12(2), 1974.

[35] A. Isidori. *Nonlinear Control Systems.* Springer Verlag, New York, 3 edition, 1995.

[36] A. Jadbabaie and J. Hauser. On the stability of receding-horizon control with a general terminal cost. *IEEE Transactions on Automatic Control*, 50(5):674–678, 2005.

[37] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding-horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46(5):776–783, 2001.

[38] H. K. Khalil. *Nonlinear Systems.* Prentice Hall, Englewood Cliffs, NJ, 2 edition, 1995.

[39] E. B. Lee and L. Markus. *Foundations of Optimal Control Theory.* John Wiley & Sons, New York, 1989.

[40] J. Lin, A. S. Morse, and B. D. O. Anderson. The multi-agent rendezvous problem. In *IEEE Conf. on Decision and Control*, pages 1508–1513, Maui, HI, December 2003.

[41] Z. Lin, M. Broucke, and B. Francis. Local control strategies for groups of mobile autonomous agents. *IEEE Transactions on Automatic Control*, 49(4):622–629, 2004.

[42] D. G. Luenberger. *Optimization by Vector Space Methods.* John Wiley, New York, 1969.

[43] N. A. Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers, San Mateo, CA, 1997.

[44] L. Marconi, A. Isidori, and A. Serrani. Autonomous vertical landing on an oscillating platform: an internal model based approach. *Automatica*, 38:21–32, 2002.

[45] J. A. Marshall, M. E. Broucke, and B. A. Francis. Formations of vehicles in cyclic pursuit. *IEEE Transactions on Automatic Control*, 49(11):1963–1974, 2004.

[46] P. Martin, S. Devasia, and B. Paden. A different look at output tracking: control of a VTOL aircraft. *Automatica*, 32(1):101–107, 1996.

[47] S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli. On synchronous robotic networks – Part I: Models, tasks and complexity notions. & Part II: Time complexity of rendezvous and deployment algorithms. *IEEE Transactions on Automatic Control*, April 2005. To appear. Short versions were presented at the 2005 CDC/ECC in Seville, Spain.

[48] S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli. On synchronous robotic networks – Part I: Models, tasks and complexity notions. & Part II: Time complexity of rendezvous and deployment algorithms. *IEEE Transactions on Automatic Control*, April 2005. Submitted.

[49] Jirí Matousek, Micha Sharir, and Emo Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4/5):498–516, 1996.

[50] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: stability and optimality. *Automatica*, 36(6):789–814, 2000.

[51] N. Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of the Association for Computing Machinery*, 31(1):114–127, 1984.

[52] G. Notarstefano and F. Bullo. Distributed consensus on enclosing shapes and minimum time rendezvous. In *IEEE Conf. on Decision and Control*, pages 4295–4300, San Diego, CA, December 2006.

[53] G. Notarstefano, J. Hauser, and R. Frezza. Trajectory manifold exploration for the PVTOL aircraft. pages 5848–5853, Dec. 2005.

[54] W. Ren, R. W. Beard, and T. W. McLain. Coordination variables and consensus building in multiple vehicle systems. In V. Kumar, N. E. Leonard, and A. S. Morse, editors, *Cooperative Control*, volume 309 of *Lecture Notes in Control and Information Sciences*, pages 171–188. Springer Verlag, 2004.

[55] L. M. Silverman and B. D. O. Anderson. Controllability, observability and stability of linear systems. *SIAM Journal on Control*, 6(1), 1968.

[56] E. D. Sontag. Universal nonsingular controls. *Systems and Control Letters*, 19:221–224, 1992.

[57] E. D. Sontag. Control of systems without drift via generic loops. *IEEE Transactions on Automatic Control*, 40:1210–1219, 1995.

[58] D. P. Spanos and R. M. Murray. Motion planning with wireless network constraints. In *American Control Conference*, pages 87–92, Portland, OR, June 2005.

[59] M. W. Spong and D.J. Block. The pendubot: a mechatronic system for control research and education. In *IEEE Conf. on Decision and Control*, pages 555–556, Dec. 1995.

[60] H. J. Sussmann. Single-input observability of continuous-time systems. *Mathematical Systems Theory*, 12:371–393, 1979.

[61] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal on Computing*, 28(4):1347–1363, 1999.

[62] P. Tabuada, G. J. Pappas, and P. Lima. Motion feasibility of multi-agent formations. *IEEE Transactions on Robotics*, 21(3), 2005.

[63] H. G. Tanner, G. J. Pappas, and V. Kumar. Leader-to-formation stability. *IEEE Transactions on Robotics and Automation*, 20(3):443–455, 2004.

[64] M. M. Zavlanos and G. J. Pappas. Controlling connectivity of dynamic graphs. In *IEEE Conf. on Decision and Control and European Control Conference*, pages 6388–6393, Seville, Spain, December 2005.

[65] Eberhard Zeidler. *Applied Functional Analysis: Main Principles and their applications.* Springer-Verlag, New York, 1995.