Order no.: **2012-19-TH**

THÈSE DE DOCTORAT

DOMAINE : STIC

SPECIALITE : AUTOMATIQUE

Ecole Doctorale « Sciences et Technologies de l'Information des Télécommunications et des Systèmes »

Présentée par :

Ionela PRODAN

Sujet :

Commande sous contraintes de systèmes dynamiques multi-agents

Soutenue le 3/12/2012

devant les membres du jury :

M. Georges BITSORIS	U. Patras, Grèce	Invité
M. Patrick BOUCHER	SUPELEC, France	Examinateur
Mme. Eva CRUCK	DGA, France	Examinatrice
M. Fernando FONTES	FEUP, Portugal	Invité
M. Morten HOVD	NTNU, Norvège	Rapporteur
M. Stéphane LE MENEC	EADS, France	Invité
M. Rudy NEGENBORN	TU Delft, Pays Bas	Examinateur
M. Silviu-Iulian NICULESCU	CNRS-SUPELEC, France	Co-encadrant
M. Sorin OLARU	SUPELEC, France	Directeur de thèse
M. Fernando PEREIRA	FEUP, Portugal	Rapporteur
Mme. Cristina STOICA	SUPELEC, France	Co-encadrante

Thanks

The work presented in this thesis was conducted as a collaboration between the Automatic Department of Supélec and the Laboratory of Signals and Systems of CNRS-Supélec, with funding from the EADS Foundation. During these last three years I have succeeded to accomplish my objectives due to the help, support and friendship of many people. I would like to take this opportunity to express again all my gratitude to each one of them.

First of all, I wish to thank my supervisor, Professor Sorin Olaru for his effort in guiding me in all research issues, his enthusiasm for research and his academic support. To him and to my co-supervisors Assistant Professor Cristina Stoica and Professor Silviu-Iulian Niculescu I am grateful for their help and continuous encouragement to explore new research challenges and pursue them successfully.

I thank to Professor Patrick Boucher, the former Head of the Automatic Department for welcoming me in the laboratory and for making sure that my thesis is developing in an optimal way. Beyond these aspects, I am honored by the interest he has always shown for my work and his presence in my thesis committee as the president of the jury. I am thankful to Professor Didier Dumur, the current Head of the Automatic Department and all the professors in the department for their constant support. Furthermore, I would like to thank Ms Marie-Claire Certiat who has regularly attended my work as representative of the EADS Foundation and for her encouragement.

My deepest appreciation goes also to all the other members that agreed to be part of the committee: Professor Morten Hovd, Professor Fernando Pereira, Research Engineer Eva Cruck, Associate Professor Rudy Negenborn, Professor Georges Bitsoris, Professor Fernando Fontes and Research Engineer Sthépane Le Menec. In particular, I am grateful

to the reviewers Professor Morten Hovd and Professor Fernando Pereira for their careful reading of the manuscript and all the provided remarks which helped in improving the overall quality of the exposition.

During these three years I had the chance to work with a large number of people and to visit different laboratories. In particular, I thank Professor Georges Bitsoris, Professor João Sousa, Dr. Ricardo Bencatel, Professor Fernando Fontes for the fruitful discussions I had with them. Also, I enjoyed working with Dr. Florin Stoican and I have benefited greatly from the clarity of vision he brings to his work. I am most grateful for the time I spent at the Underwater Systems and Technology Laboratory from the University of Porto in Portugal, a collaboration initiated by my supervisor. I enjoyed there everyone's friendliness and support, in particular Catarina Morais who I thank for giving up a few hours from her schedule to show me the beautiful city of Porto. Thanks to Professor João Sousa I had the chance to go at the Portuguese Air Force Academy, OTA, for the flight experiments, which proved to be a unique experience for me. Also, I would like to thank Professor Anders Rantzer for giving me the privilege to spend some time in the Automatic Department from University of Lund in Sweden.

When I look back during my scientific study, I am grateful to many, many people who have given me challenges and believed in my competence. My thanks go to Professor Cristian Oara and Professor Radu Stefan from University "Polithenica" of Bucarest, as well as to my highschool math professor Iuliana Turcu.

My warmest thanks go to my friends and colleagues: Vali, Catalin, Emanuel, Georgi, Florin, Ana, Warody, Safta, Anca, Nam, Nikola, Andreea, Daniel, Julien, Antoine, Christophe, Stefan, Dorin, Adriano, Raluca, Bogdan, Cristina, Ali.

I would not be here if it wasn't for my parents, to whom I owe the greatest debt of gratitude. I am also blessed to have my sisters Daniela and Iuliana, my life together with them is an enormous joy, and it is, above all, my greatest source of strength. This work is dedicated to them.

Pentru surorile mele Daniela si Iuliana,
parintilor mei.
******

"Satisfaction lies in the effort, not in the attainment; full effort is full victory."

Mahatma Gandhi

ÉCOLE SUPÉRIEURE D'ÉLECTRICITÉ

Abstract

SPECIALITE : AUTOMATIQUE

Doctorate degree

by Ionela Prodan

The goal of this thesis is to propose solutions for the optimal control of multi-agent dynamical systems under constraints. Elements from control theory and optimization are merged together in order to provide useful tools which are further applied to different problems involving multi-agent formations.

The thesis considers the challenging case of agents subject to dynamical constraints. When designing control decisions, it becomes thus natural to take into account not only exogenous factors (e.g., obstacles, reference tracking, etc.) but also the internal (state) dynamics of the agents and their structural properties (as settle-time or nonholonomic characteristics).

To deal with these issues, well established concepts like set-theory, differential flatness, Model Predictive Control (MPC), Mixed-Integer Programming (MIP) are adapted and enhanced. Using these theoretical notions, the thesis concentrates on understanding the geometrical properties of the multi-agent group formation and on providing a novel synthesis framework which exploits the group structure. In particular, the formation design and the collision avoidance conditions are casted as geometrical problems and optimization-based procedures are developed to solve them. Moreover, considerable advances in this direction are obtained by efficiently using MIP techniques (in order to derive an efficient description of the non-convex, non-connected feasible region which results from multi-agent collision and obstacle avoidance constraints) and stability properties (in order to analyze the uniqueness and existence of formation configurations).

The existence and uniqueness of a tight formation of agents is further linked with constraints over the eigenstructure of the state matrices of the agents.

Lastly, some of the obtained theoretical results are applied on a challenging practical application. A novel combination of MPC and differential flatness (for reference generation) is used for the flight control of Unmanned Aerial Vehicles (UAVs).

ÉCOLE SUPÉRIEURE D'ÉLECTRICITÉ

Résumé

SPECIALITE : AUTOMATIQUE

Diplôme de doctorat

par Ionela Prodan

L'OBJECTIF de cette thèse est de proposer des solutions aux problèmes liés à la commande optimale de systèmes dynamiques multi-agents en présence de contraintes. Des éléments de la théorie de commande et d'optimisation sont appliqués à différents problèmes impliquant des formations de systèmes multi-agents.

La thèse examine le cas d'agents soumis à des contraintes dynamiques. Il devient donc naturel de tenir compte non seulement des facteurs exogènes (par exemple les obstacles, le suivi de référence, etc.), mais aussi de la dynamique interne (l'état) d'agents et de leurs propriétés.

Pour faire face à ces problèmes, les concepts bien établis tels que la théorie des ensembles, la platitude différentielle, la commande prédictive (Model Predictive Control - MPC), la programmation mixte en nombres entiers (Mixed-Integer Programming - MIP) sont adaptés et améliorés. En utilisant ces notions théoriques, ce travail de thèse a porté sur les propriétés géométriques de la formation d'un groupe multi-agents et propose un cadre de synthèse original qui exploite cette structure.

En particulier, le problème de conception de formation et les conditions d'évitement des collisions sont formulés comme des problèmes géométriques et d'optimisation pour lesquels il existe des procédures de résolution. En outre, des progrès considérables dans ce sens ont été obtenus en utilisant de façon efficace les techniques MIP (dans le but d'en déduire une description efficace des propriétés de non convexité et de non connexion d'une région de faisabilité résultant d'une collision de type multi-agents avec des contraintes d'évitement d'obstacles) et des propriétés de stabilité (afin d'analyser l'unicité et l'existence de configurations de formation de systèmes multi-agents). L'existence et l'unicité d'une formation étroite d'agents sont, en outre, liées à des contraintes sur la structure propre des matrices d'état des agents.

Enfin, certains résultats théoriques obtenus ont été appliqués dans un cas pratique très intéressant. On utilise une nouvelle combinaison de la commande prédictive et de la platitude différentielle (pour la génération de référence) dans la commande et la navigation de véhicules aériens sans pilote (UAVs).

Systèmes dynamiques multi-agents du point de vue de la commande théorique

Tout d'abord, afin de percevoir les bénéfices retirés de l'emploi des systèmes multi-agents dans l'ingénierie de la commande, il est nécessaire de s'attarder un peu sur la définition de ces systèmes. Il existe plusieurs définitions d'un agent, ce qui est dû principalement à l'universalité de ce mot et au fait qu'il ne peut pas être propre à une seule communauté. La plupart des définitions proviennent de la communauté informatique et ils ont tous trois concepts de base : la notion d'agent, de son environnement et de son autonomie. La définition suivante est souvent citée : "un agent est une entité matérielle ou logicielle qui se trouve dans un certain environnement, et qui est capable d'actions autonomes dans l'environnement afin de répondre à ses objectifs". Comme l'indique la définition, les agents sont soit des entités physiques (matérielles), ce qui est le cas des robots, des véhicules ou des piétons, dans un environnement physique (par exemple, un système de commande), soit des entités virtuelles dans un environnement informatique (par exemple, des sources de données, des sources informatiques), ce qui est le cas des agents logiciels. Cependant, dans le contexte des agents de commande, seul le premier cas est intéressant

ici, car il y a des applications directes en ingénierie de la commande. Par agent, on désigne “tout système de commande”, comme par exemple un dispositif de thermostat qui est une instance d’un système de commande. Il est situé dans son environnement, il réagit aux changements de température de l’environnement et présente également un certain degré d’autonomie.

Les systèmes multi-agents sont des systèmes dynamiques qui se décomposent en de multiples sous-systèmes (véhicules par exemple), comprenant des capteurs et des actionneurs, qui ont la capacité de communiquer entre eux pour accomplir différentes tâches. La clé de l’étude des systèmes multi-agents réside dans la formalisation de la coordination entre les agents. Cette coordination peut être décrite soit comme une coopération entre plusieurs agents en vue d’accomplir une tâche commune, soit comme une négociation entre des agents ayant des intérêts différents. Un exemple de coopération de systèmes multi-agents est constitué par un ensemble d’avions sans pilote, exécutant des opérations de recherche et de surveillance comme des patrouilles au-dessus de zones boisées afin de détecter d’éventuels incendies.

Actuellement, de grands groupes d’agents peuvent être coordonnés et commandés de manière efficace dans l’exécution des tâches diverses, par exemple, le comportement des piétons dans une foule, problématique essentielle dans l’évaluation de la sécurité des infrastructures sociales, la commande en eau des canaux d’irrigation, la distribution de l’eau et la gestion des réseaux d’égouts, l’observation de l’espace et la fluidité de la circulation automobile.

Il est devenu évident que l’étude des systèmes dynamiques est essentielle dans le domaine de la commande des systèmes. Dans la commande des systèmes dynamiques, les décisions devraient être déterminées et mises en œuvre en temps réel. La rétroaction est largement utilisée pour faire face aux incertitudes sur le système et son environnement.

Les problèmes qui apparaissent dans le contexte de la commande coopérative de systèmes dynamiques multi-agents incluent :

- la planification de mouvements qui répond à la question classique “comment y arriver ?” ;

- la commande de la formation afin de positionner chaque agent de façon fixe par rapport à chacun des autres agents ;
- la commande centralisée, distribuée ou décentralisée qui gère le problème de la communication entre l'agent et la structure de la loi de commande ;
- la manipulation des contraintes qui doivent habituellement être intégrées dans la conception de la commande ;
- la stabilité en fonction de la nature du problème (la stabilité peut être comprise comme la stabilité de la formation, le suivi dans certaines limites ou simplement le fait que les agents restent dans une région bornée) ;
- la robustesse par rapport aux perturbations qui affectent le système et qui doivent être prises en compte ; des problématiques spécifiques multi-agents doivent également être prises en compte (par exemple, la présence d'obstacles mobiles, l'estimation de l'état des autres agents).

Une approche basée sur l'optimisation pour la commande des systèmes coopératifs

Cette thèse se propose de créer un cadre unifié pour la commande optimale de systèmes multi-agents en présence des contraintes physiques, économiques ou de communication. Les principaux objectifs de notre recherche sont :

- l'utilisation des méthodes ensemblistes pour le traitement des systèmes sous contraintes (par exemple, liées à l'évitement des collisions avec les éventuels obstacles et des collisions entre les agents) ;
- le développement de lois de commande d'un groupe d'agents afin d'accomplir un mouvement en formation (par exemple la planification et le suivi de trajectoire).

Les outils qui permettent la réalisation de ces objectifs et qui ont été utilisés dans le cadre de cette thèse sont les méthodes ensemblistes, la platitude, la programmation mixte en nombres entiers (Mixed-Integer Programming - MIP), la commande optimale

et la commande prédictive (Model Predictive Control - MPC) avec des contraintes non-convexes. Un aspect important de l'approche de la commande prédictive est caractérisé par sa capacité à gérer des contraintes génériques d'état et de commande et représente une notion prédominante dans le travail de thèse qui est présenté dans ce manuscrit. La commande à horizon glissant étant une technique basée sur l'optimisation représente un candidat naturel à la résolution problèmes associés aux systèmes multi-agents. Cette technique peut viser des objectifs très généraux, comme par exemple, la convergence vers une formation, le suivi d'une trajectoire, tout en prenant en compte les contraintes anti-collisions.

Les systèmes dynamiques considérés, qui sont utilisés via leurs modèles dans un but de prédiction, peuvent être à temps continu ou discret dans le temps, homogènes ou hétérogènes, linéaires ou non-linéaires, et avec ou sans contraintes. Dans ce manuscrit, nous travaillons la plupart du temps avec une dynamique généralement héritée de la dynamique du véhicule, simplifiée dans le cas des (sous-) systèmes, Ce sont des modèles de type double intégrateur, linéaires invariants dans le temps et non linéaires monocycles, mis en présence de perturbations bornées et de contraintes. Il faut noter que nous avons choisi les dynamiques simplifiées et leurs analogies afin de faciliter l'interprétation des résultats numériques. Le traitement des contraintes, la conception de la commande et les techniques résultantes sont valables pour des formulations générales de la dynamique et des contraintes.

Pour les tâches impliquant une coopération entre un groupe d'agents, le suivi de trajectoire est souvent crucial dans la réalisation de l'objectif de coopération. Il arrive souvent qu'une trajectoire de référence doive être pré-calculée. Quand on parle de signaux de références, il est important de souligner la différence entre deux notions proches : trajectoire de référence et chemin de référence. Le chemin de référence prévoit seulement une route souhaitée pour les agents pour lesquels il ne peut pas exister d'entrée possible. En ce sens, la trajectoire de référence est supérieure parce qu'elle permet à la fois une entrée et un état faisable. Son inconvénient réside dans sa dépendance temporelle qui impose souvent une contrainte supplémentaire sur le fonctionnement en temps réel, tandis que le chemin de référence reste indépendant du temps. Bien sûr, trouver des références d'entrée et des signaux d'état faisables est généralement une tâche difficile. Nous avons choisi d'utiliser dans le présent manuscrit un outil générique, celui basé sur la

platitude différentielle pour la construction d'une trajectoire de référence. Les systèmes différentiellement plats sont bien adaptés aux problèmes nécessitant une planification de trajectoires. Un des intérêts de la platitude est qu'elle simplifie le problème de génération de trajectoire, en le ramenant à la recherche d'une trajectoire de sortie plate (par le fait que la sortie plate décrit complètement le comportement du système).

Si on accepte la commande MPC comme cadre de conception et si on dispose d'une trajectoire de référence, on peut en principe concevoir une boucle de régulation nominale, mais la robustesse doit être prise en compte par des outils d'analyse et des notions spécifiques. Pour une approche robuste, nous utilisons la théorie des ensembles. Chaque fois que les bruits et les perturbations sont bornés nous chercherons à construire des ensembles invariants bornés qui caractérisent la dynamique du système. Par exemple, nous pouvons construire des régions de sécurité autour des agents afin de garantir l'évitement des collisions pour toutes les valeurs des bruits bornés. Pour des raisons numériques, les ensembles que nous choisissons sont des ensembles polyédraux. Ils offrent un excellent équilibre entre flexibilité de la représentation et facilité de mise en œuvre numérique des algorithmes. Plus précisément, nous allons les utiliser dans le cadre de systèmes multi-agents pour décrire des régions de sécurité, des obstacles et des régions de faisabilité. Une notion connexe, la norme polyédrale, sera également utilisée afin de construire ce qu'on appelle une fonction polyédrale et une fonction somme, qui vont constituer la base pour la construction de fonctions d'exclusion (ou fonctions de pénalité).

Contribution à la description de régions de faisabilité non-convexes via la programmation mixte en nombres entiers

L'évitement des collisions est souvent le problème le plus difficile dans le contexte de la gestion des agents multiples, car certaines contraintes (statiques ou dynamiques) sont non-convexes. Cela arrive parce que l'évolution d'un système dynamique, dans un environnement présentant des obstacles, ne peut être modélisée que sous forme de régions non-convexes possibles, c'est-à-dire que la trajectoire de l'agent doit éviter une région convexe (union de régions convexes) représentant un obstacle (contraintes statiques) ou un autre agent (contraintes dynamiques - menant à un paramétrage de l'ensemble des contraintes par rapport à l'état actuel).

La plus grande partie de la littérature traite du problème d'évitement des collisions des systèmes multi-agents punctiformes, un cas éloigné de la réalité. En effet, dans beaucoup d'applications, le positionnement relatif entre les agents est important : par exemple, dans la gestion du trafic aérien, deux avions ne sont pas autorisés à s'approcher l'un de l'autre à moins d'une distance de sécurité spécifique. Afin de prendre en compte ce degré supplémentaire de difficulté, nous avons proposé d'associer à chaque agent punctiforme une région de sécurité qui permet la formulation de contraintes non-convexes d'évitement des collisions. Nos études ont ainsi conduit à de nouvelles méthodes permettant de résoudre des problèmes d'optimisation sous contraintes qui apparaissent dans la commande des systèmes multi-agents.

Du point de vue de la mise en œuvre, les solveurs purs d'optimisation non convexe peuvent être utilisés pour les problèmes de commande évoqués ici, mais, à notre avis, sont sujets à des erreurs numériques et difficiles à manipuler. Les études effectuées ont montré que les techniques dites de "programmation mixte en nombres entiers" (Mixed Integer Programming - MIP) sont appropriées dans le traitement des problèmes d'optimisation sous contraintes non convexes. Malgré leurs capacités de modélisation, des difficultés liées au traitement numérique sont souvent rencontrés dans la résolution des problèmes MIP. Des travaux existent dans la littérature, qui permettent de réduire les exigences de calcul des méthodes de programmation mixte en nombres entiers, afin de rendre ces méthodes plus attractives en ce qui concerne les applications en temps réel. Dans ce cadre, nous avons développé une technique originale afin de reformuler le problème sous contraintes linéaires. Cette nouvelle formulation réduit le nombre de variables binaires nécessaires à la description unitaire des ensembles convexes non-connectés (ou de leurs compléments) en utilisant des variables binaires auxiliaires. De plus, l'approche proposée a été étendue au traitement des régions non connectées non convexes. Les principaux apports de ce travail mettent en relief quelques-uns des aspects importants de l'approche, soit :

- une représentation convexe dans l'espace étendu d'état ainsi que des variables binaires en utilisant la notion d'hyperplan associé ;
- une réduction de la complexité du problème en utilisant des techniques d'algèbre booléenne (à cause de la complexité du problème, il faut seulement un nombre polynomial de sous-problèmes (Programmation Linéaire (LP) ou Programmation

Quadratique (QP) des problèmes) qui doivent être résolus avec des avantages évidents dans l'effort de calcul) ;

- une propriété remarquable d'association optimale entre les régions et la représentation binaire conduit à la minimisation du nombre de contraintes.

La commande de formation des systèmes dynamiques multi-agents

La commande de formation d'agents est devenue l'un des problèmes bien connus dans les systèmes multi-agents. Par rapport à un seul agent, un groupe d'agents travaillant ensemble présente de nombreux avantages ; cela a été démontré dans de nombreuses applications impliquant la commande des systèmes coopératifs. La formation est un cas particulier de coopération. Dans notre cadre d'étude, nous proposons la définition suivante d'une formation : une formation est une organisation d'un groupe d'agents se déplaçant ensemble sous certaines contraintes, dans un espace restreint, avec des objectifs communs.

La caractérisation et la convergence vers une formation constituent des questions classiques pour la commande des systèmes coopératifs. En outre, le problème spécifique de maintenir une formation devient encore plus difficile si l'on doit faire en sorte que tous les agents évitent les collisions au sein du groupe et/ou avec des obstacles.

La contribution apportée par cette thèse vise à améliorer la compréhension des propriétés géométriques et permet de dessiner un cadre de synthèse novateur (à notre connaissance) exploitant la structure géométrique proposée. La conception de formation et les conditions d'évitement des collisions sont formulées comme des problèmes géométriques et des procédures d'optimisation sont élaborées pour les résoudre. De plus, nous avons obtenu des avancées considérables dans ce sens en utilisant efficacement les techniques MIP (afin d'en tirer une description cohérente de la région de faisabilité dans l'espace des solutions) et les propriétés de stabilité (pour analyser le caractère unique et l'existence de configurations de formation).

Nous nous concentrons sur la commande fondée sur l'optimisation de plusieurs agents ayant chacun une dynamique autonome, ainsi qu'un objectif global à atteindre, comme une formation étroite avec les spécifications désirées et les comportements sans collision. Pour réduire le temps de calcul, nous utilisons le comportement "nominal" des agents et nous considérons des régions de sécurité autour d'eux afin de compenser les effets des perturbations affectant les systèmes "réels". En outre, ces régions sont définies en utilisant la théorie des ensembles invariants pour éviter de devoir les recalculer pendant le fonctionnement en temps réel. Il faut noter que ce choix garantit également un degré de robustesse certain, en dépit du fait que la commande en temps réel est réalisée en utilisant des modèles de prédiction nominaux.

L'approche proposée consiste à décomposer le problème de commande de la formation en deux problèmes distincts.

- Tout d'abord, la configuration idéale est définie hors ligne. Une configuration minimale est déterminée par rapport à une fonction de coût proposée avec des contraintes imposées par les régions de sécurité. Une contribution particulière, ici, est l'introduction d'une contrainte de point fixe supplémentaire (à savoir, les positions cibles sont aussi des points d'équilibre pour la dynamique considérée) afin d'assurer la convergence vers la configuration prédéfinie.
- Ensuite, en temps réel, une optimisation, avec un horizon glissant, est utilisée conjointement avec l'attribution des tâches relatives à la configuration minimale.

La commande en temps réel est alors fondée sur une procédure en deux étapes.

- La première étape est de déterminer où doit aller chaque agent dans la formation. Celle-ci est équivalente à la résolution d'un problème standard d'affectation de tâches. Ce problème n'est pas nouveau dans le domaine de l'optimisation combinatoire, cependant, dans le cadre de la commande des systèmes multi-agents, l'affectation optimale en temps réel apporte une nouvelle dimension (dynamique) en intégrant des cibles variables dans le temps.
- La seconde étape est de résoudre un problème d'optimisation mixte en nombres entiers en fonction de la géométrie de la formation prédéfinie et des régions de

sécurité associés, afin d'obtenir des lois de commande efficaces pour les agents individuels.

Enfin, ces deux problèmes différents sont incorporés dans la commande prédictive, ce qui conduit à un problème d'optimisation destiné à guider le groupe d'agents vers une formation prédéfinie avec des cibles associées.

Un autre point abordé dans ce manuscrit a été le problème de la commande prédictive (centralisée, distribuée et décentralisée) de systèmes multi-agents en formation. Malgré toutes les améliorations de la formulation MIP, le problème de la commande prédictive centralisée pour les systèmes multi-agents est toujours difficile à résoudre. Afin de proposer une solution plus performante, la solution MIP a été étendue dans le contexte de la commande prédictive distribuée. Plus précisément, le groupe d'agents a d'abord été partitionné en voisinages et ensuite une description MIP de la région de faisabilité, dans laquelle se trouve chaque agent, a été élaborée. Avec une communication adéquate entre les sous-groupes d'agents voisins, les demandes de performance et de stabilité imposées par le cahier des charges peuvent être respectées. Pour ces systèmes multi-agents, la charge de calcul est réduite considérablement dans le cas de la commande prédictive distribuée basée sur des techniques MIP, en comparaison avec les formulations plus classiques de la commande prédictive..

Pour offrir une alternative à l'utilisation de techniques MIP dans ce projet, nous avons également exploré des stratégies améliorées de commande prédictive décentralisée basées sur la méthodologie du champ de potentiel. Cette approche a été utilisée afin de synthétiser une loi de commande prédictive uniquement à partir des informations locales au sein du groupe d'agents. L'utilisation de différents champs de potentiel, attractifs ou répulsifs, autour des agents, permet à la fois de garder les agents en formation et d'éviter les collisions entre eux. Cette approche a l'avantage de réduire la complexité de calcul par rapport aux techniques précédentes basées sur les techniques MIP. En revanche, la preuve formelle de stabilité de la formation des systèmes multi-agents s'avère difficile si on utilise des techniques de différents champs de potentiel.

Les évaluations de la commande sous contraintes de systèmes dynamiques multi-agents

Un autre aspect théorique traité dans cette thèse est la description du comportement limite d'un agent en présence de contraintes adverses. Plus précisément, ce type de contraintes rend la convergence de la trajectoire d'un agent vers l'origine[1] impossible à réaliser. Á notre connaissance, dans la littérature il n'existe pas de résultats traitant le cas de contraintes qui ne sont pas satisfaites lors de la convergence vers l'origine. Nous avons ainsi développé une stratégie de commande duale afin que l'agent converge vers un point fixe unique et stable qui se trouve sur la frontière de la région interdite[2], et non pas vers l'origine.

Par conséquent, dans un premier temps, nous effectuons une analyse détaillée du comportement limite pour un système dynamique linéaire en présence de contraintes géométriques adverses. Plus précisément, il nous faut définir les points fixes et les propriétés d'invariance de la trajectoire d'état du système, tout en évitant une région convexe contenant l'origine strictement à l'intérieur de cette région. Dans le contexte de systèmes multi-agents, cette région peut, en fait, représenter un obstacle (contraintes statiques), mais peut être étendue à la zone de sécurité d'un autre agent (conduisant à une paramétrisation de l'ensemble des contraintes à l'égard de l'état global actuel).

Dans un deuxième temps, nous cherchons à assurer la stabilité dans la région de faisabilité de l'espace d'état en utilisant une stratégie de commande duale. Les principes fondamentaux sont ceux de la commande prédictive y compris les contraintes d'évitement de collision. Les liaisons entre points fixes, invariance et lois de commande affines nous permettent d'offrir les conditions nécessaires et suffisantes pour l'existence d'un point d'équilibre stable ayant toute la région de faisabilité comme bassin d'attraction.

Nous avons ensuite étendu cette analyse pour une formation multi-agents. La condition d'unicité dès les derniers points fixes se traduit dans la formulation multi-agents par une condition de configuration unique. Il est important de s'assurer qu'une loi de commande

[1] Nous considérons par la suite que l'origine est le point d'équilibre du système dynamique à commander. L'existence d'une contrainte adverse signifie que ce point n'est pas accessible au système considéré.

[2] La région interdite à l'agent représente une région polyèdrale qui contient le point d'équilibre naturel.

ne conduira pas à un comportement cyclique pour les agents, ce qui impliquerait une surconsommation d'énergie. Formellement, le fait qu'un ensemble d'agents reste dans (ou converge vers) une configuration unique et stable, à la suite d'une stratégie de commande appropriée, est équivalent à dire que dans un espace étendu, il existe de façon unique un point qui peut être fixé par la même stratégie de commande. Par conséquent, nous pouvons garantir la stabilité d'une formation multi-agents, mais bien sûr, cela dépend de la façon le problème d'optimisation est résolu.

Il existe de nombreuses applications à ce travail présentant un intérêt particulier, à savoir celles où les contraintes statiques ou dynamiques doivent être respectées. Parmi celles-ci on peut citer en exemple la commande et la coordination d'une plateforme mobile sur l'océan. Les modules homogènes de base doivent être en mesure d'accomplir le maintien du poste sur la mer à long terme, en présence de vagues, de vents et de courants. Par conséquent, les modules indépendants doivent être commandés de manière efficace pour maintenir le poste aligné. Cette tâche peut être facilement accomplie si chaque module converge vers différents points fixes convenablement choisis. Par ailleurs, il est bien connu que l'Atlantique Nord est l'un des environnements les plus inhospitaliers de la planète. Pourtant, c'est ici que le pétrole (au large des côtes de la Norvège notamment) et le gaz ont été exploités pendant des années avec un succès remarquable. Afin d'assurer sa prospérité dans des conditions difficiles, l'industrie offshore repose sur une variété de technologies innovantes. Un exemple est le vaisseau cargo brise-glace et/ou camion-citerne qui a besoin de briser la glace autour d'une plateforme. Par conséquent, le navire brise-glace doit manœuvrer le plus près possible de la plateforme, tout en évitant la collision avec celle-ci (c'est-à-dire que le navire doit converger vers un cycle limite).

Application réelle - commande de vol de véhicules aériens sans pilote

Nous avons appliqué et validé les méthodes théoriques proposées dans le suivi de trajectoire de plusieurs drones (Unmanned Aerial Vehicles - UAVs) en utilisant le matériel mis à notre disposition par le Laboratoire des Systèmes et de la Technologie Sous-marine (LSTS), de l'Université de Porto, au Portugal, au sein duquel j'ai été invitée à séjourner pendant deux mois.

Les drones sont des appareils volants sans pilote qui ont beaucoup attiré l'attention ces dernières années, notamment après les tragiques événements survenus le 11 septembre 2001, lorsque le monde entier a remis en question la sécurité à bord des avions et d'autres moyens de transport aérien. Actuellement, la plupart des drones sont utilisés dans la planification de missions militaires de soutien de combat sans pilote, la surveillance du trafic, la surveillance et la recherche de survivants. La gamme des applications civiles possibles en matière de drones est actuellement en train de s'étendre. Les nombreuses technologies développées pour les drones militaires sont de plus similaires, voire même identiques, à celles requises pour les drones civils.

L'un des défis principaux de la recherche est d'améliorer et d'augmenter leur autonomie, en particulier de permettre l'application de méthodes avancées de commande. Celles-ci doivent respecter les contraintes de la dynamique du véhicule et de permettre la reconfiguration de la trajectoire du véhicule dans le cas où des événements inattendus se produiraient dans le système. L'utilisation combinée de la commande prédictive avec les concepts de la platitude représente une combinaison importante dans l'état de l'art, laquelle permet de gérer le contrôle en temps réel, la génération de trajectoire, et peut faire face aux problèmes de robustesse en utilisant la théorie des ensembles. Par ailleurs, à notre connaissance, une telle stratégie n'a pas été évaluée dans des essais en vol sur des véhicules réels.

Au cours des essais sur les drones mis à notre disposition au Portugal, nous avons testé les différents algorithmes et techniques développés pendant ce projet et les résultats des tests réels se sont révélés tout à fait satisfaisants.

Les essais en vol ont eu lieu à la Base Aérienne Portugaise OTA en mai 2012 sur les plateformes appartenant à l'Air Force Académie et le LSTS, laboratoire de l'Université de Porto. Ainsi, en utilisant des lois de commande prédictive élaborées au cours de cette thèse et des techniques de platitude différentielle[3], l'objectif a été de commander un drone afin de respecter des points de passage imposés. Pour chaque drone testé, nous avons, tout d'abord, généré une trajectoire plate[4] passant par les points de passage

[3]Les techniques basées sur la platitude ont permis de générer des trajectoires entre un point de départ et un point d'arrivée, en passant par plusieurs points de passage imposés par le cahier de charge.

[4]Les drones sont des systèmes différentiellement plats donc bien adaptés aux problèmes nécessitant une planification de trajectoires. Un des intérêts de la platitude est de simplifier le problème de génération de trajectoire, en le ramenant à la recherche d'une trajectoire de la sortie plate par le fait que la sortie plate décrit complètement le comportement du système.

intermédiaires et ensuite nous avons commandé le drone afin de suivre la trajectoire imposée en utilisant les techniques de commande la prédictive.

Conclusions

En ce qui concerne les travaux connexes, nous avons constaté une grande variation dans la définition de la notion d' "agent". Les entités hétérogènes dans les systèmes coopératifs pourraient être aussi simples que les agents scalaires sans aucune dynamique ou bien une dynamique du second ordre au point de masse des modèles ou même une dynamique non linéaire avec contraintes de mouvement non holonomes. Pour notre étude, nous avons considéré le cas plus difficile d'agents avec une dynamique, puisque nous nous intéressons à la classe d'applications pertinentes dans un contexte de commande. Inutile de dire que cela signifie que les décisions de commande doivent tenir compte, non seulement des facteurs exogènes (par exemple, la présence d'obstacles, suivi de références, etc.), mais également de la dynamique interne (état) des agents et de leurs contraintes dynamiques (comme le réglage des temps, des caractéristiques non holonomes ou des retards de transmission).

Ayant appliquée la théorie ensembliste aux systèmes multi-agents, il était naturel d'envisager des outils spécifiques dans ce contexte. Par exemple, les obstacles et les agents eux-mêmes ont été décrits comme des ensembles (généralement convexes) et la collision et les contraintes d'évitement d'obstacles ont été formulées avec les opérateurs ensemblistes comme l'inclusion d'un élément dans un ensemble ou l'intersection entre les ensembles. Rappelons également que nous avons fait une série d'hypothèses qui, sans des notions telles que la région de sécurité, auraient perdu leur sens. En particulier, nous avons supposé que les perturbations et les incertitudes peuvent être délimitées. Même s'il est parfois nécessaire de fournir une limite théorique pour les bruits et les perturbations agissant sur un système, dans la pratique, il est naturel de considérer de telles bornes ainsi que des contraintes dures.

L'utilisation de la théorie des ensembles est non seulement un moyen pratique de décrire les contraintes et les comportements, mais elle mène aussi à des résultats intéressants en soi. Par exemple, l'une des notions fondamentales de la théorie des ensembles est l'invariance d'un ensemble. Nous avons utilisé cette notion pour décrire les régions

invariantes de sécurité autour d'agents touchés par des perturbations. Ainsi, nous avons évité de devoir recalculer l'ensemble à chaque pas de temps (comme c'est généralement le cas dans la littérature).

Un autre élément important de notre approche, les techniques de programmation mixte en nombres entiers, nous a permis de décrire sous une forme exploitable la région de faisabilité non-convexe et non connectée, résultant des contraintes d'évitement d'obstacles et des autres agents. Des éléments tels que les arrangements d'hyperplans et la fusion de cellules ont permis de décrire efficacement une telle région de faisabilité. Nous croyons que les résultats et les idées sur la description d'une région de faisabilité non-convexe sont utiles non seulement pour le sujet multi-agents, mais aussi dans le cadre plus général des problèmes d'optimisation non convexes avec contraintes.

Nous avons également lié l'existence et l'unicité d'une formation étroite des agents à des contraintes sur la structure propre des matrices d'état des agents. Cette analyse formelle nous a permis de formuler d'intéressantes remarques et montre le lien profond entre des notions telles que les cycles limites, les points fixes et la conception de la formation et la commande. Il est important de souligner que ces propriétés dépendent de la stratégie d'optimisation utilisée, mais du moins dans le cas centralisé (et, dans certaines conditions, dans le cas distribué), elles peuvent être garanties.

Enfin, nous avons appliqué certains de ces résultats théoriques à une application pratique. Nous avons évalué une combinaison de la commande prédictive et de la platitude différentielle pour la commande de vol de véhicules aériens sans pilote (UAV). Dans une première étape, les contrôleurs de vol MPC ont été affinés et testés en simulation tandis que dans une deuxième étape, nous avons exécuté des essais en vol avec des drones réels (lors d'une visite au laboratoire de LSTS à Porto). Nous signalons qu'il n'est pas facile de passer de la conception d'un correcteur sur papier à la mise en œuvre en temps réel. Par conséquent, nous croyons que les résultats en temps réel sont extrêmement prometteurs et méritent une recherche approfondie.

Perspectives et directions futures

D'autres directions envisagées impliquent d'autres approches théoriques devant être étudiées et comparées, notamment en termes de robustesse face à un environnement difficile.

Dans la littérature, résoudre des problèmes d'optimisation dans des régions non convexes n'est pas une question nouvelle et nous avons montré que les techniques de programmation mixte en nombre entiers constituent l'un des meilleurs moyens de faire face à ce type de problème. Avec toutes les améliorations importantes que nous avons réalisées, la complexité de calcul est encore très dépendante de la formulation MIP et limite son utilisation pour les problèmes de taille relativement petite. Nous croyons que nous pouvons aller plus loin dans la promotion de la nouvelle interprétation géométrique des contraintes non convexes, à l'origine les problèmes rencontrés avec les systèmes multi-agents. La concentration sur les manières compactes de décrire la région de faisabilité permettra d'éviter de faire des détours inutiles en direction de la formulation "entier-mixte". En particulier, nous voulons éviter de décomposer explicitement l'arrangement des hyperplans dans des cellules car cette opération entraine une charge importante de calcul.

Nous pensons également que des améliorations significatives peuvent être apportées dans l'étude de l'existence et de l'unicité des formations multi-agents. En particulier, nous aimerions approfondir les liens entre l'assignation de structure propre, induisant des contraintes d'invariance, et le comportement limite du point de vue de la formation multi-agent. Par exemple, dans le présent manuscrit nous nous sommes concentrés sur l'existence et l'unicité d'un point fixe. Cela peut effectivement ne pas être possible dans certains cas (un drone ne peut pas avoir une position stable en vol). Dans un tel cas, nous aimerions pouvoir déterminer un cycle limite (optimal ou économique) et en déduire les propriétés qui le caractérisent : unicité, bassin d'attraction, stabilité, etc.

Pendant les essais expérimentaux en Portugal, nous avons remarqué l'influence du retard de communication sur la boucle de commande. Ainsi, une perspective intéressante s'avère être la prise en compte de ce retard de communication dans la loi de commande. Une autre piste intéressante est de faire voler plusieurs drones en formation afin de valider toutes les théories développées au cours de ce travail de thèse.

Contents

List of Figures

List of Tables

List of Algorithms

Notation

The conventions and the notations used in the manuscript are classical for the control literature. A detailed description is provided in order to increase the coherence among chapters and facilitate comparing the techniques discussed.

Let $\mathbb{R}$, $\mathbb{Z}$ and $\mathbb{N}$ denote the field of real numbers, the set of integers and the set of non-negative integers, respectively. Notations $\mathbb{R}^n$ and $\mathbb{R}^{m\times n}$ denote the vector field and the matrix field of real numbers, respectively. The same notation adopted for the sets of integer and non-negative integers.

For the signals manipulated in the manuscript the index t is used for continuous time and the index k, for discrete time, respectively. For the time index as parameter of variable, bracketed behind variable is used, e.g., $x(k) \in \mathbb{R}^n$.

Absolute values and vector inequalities are considered elementwise (unless otherwise explicitly stated), that is, $|T|$ denotes the elementwise magnitude of a matrix T and $x \leq y$ ($x < y$) denotes the set of elementwise (strict) inequalities between the components of the real vectors x and y. The ceiling value of $x \in \mathbb{R}$, denoted by $\lceil x \rceil$, is the smallest integer greater than x.

For a set $S \in \mathbb{R}^n$ we denote with $\bar{s} = \max_{s\in S} s$ the elementwise maximum, where each element is computed as $\bar{s}_i = \max_{s\in S} \ s_i$. In addition, the elementwise minimum, $\underline{s} = \min_{s\in S} s$, is defined in a similar way. $\|z\|_M$ is the weighted Euclidean norm, i.e., $\sqrt{(z^T M z)}$.

For a matrix $A \in \mathbb{R}^{n\times m}$ and a set $S \subseteq \mathbb{R}^m$, we define the set:

$$AS = \{z \in \mathbb{R}^n : \ z = Ax \ \text{ for some } \ x \in S\}.$$

The closure of a set S is denoted by $cl(S)$.

$\mathbb{B}_p^n = \{x \in \mathbb{R}^n : \|x\|_p \leq 1\}$ denotes the unit ball of norm p, where $\|x\|_p$ is the p-norm of vector x. The notation B_∞^n represents the ∞-norm ball in $\mathbb{R}^n$ of radius one ($p = \infty$ in $\mathbb{B}_p^n$). In addition, given a compact set $S \subset \mathbb{R}^n$, $B_\infty^n(S)$ denotes the set of the form $B_\infty^n(S) = \{x : \ \underline{s} \leq x \leq \bar{s}\}$, where the vector $\underline{s}$, respectively $\bar{s}$, is the elementwise

minimum, respectively maximum, of S defined above (note that $B_\infty^n(S)$ is the "smallest box" containing S).

Notations $lp(n,d)$ and $qp(n,d)$ represent the complexity of solving a linear program, quadratic program respectively, with n constraints and d variables.

The collection of all possible N combinations of binary variables will be denoted by:

$$\{0,1\}^N = \{(b_1,\ldots,b_N):\ b_i \in \{0,1\}, i=1,\ldots,N\}.$$

For a binary signal f with values in $\{0,1\}$, notation $\bar{f}$ denotes $\bar{f} = 1 - f$.

e_i denotes the i^{th} standard basis vector.

A Voronoi region, $\mathcal{V}_i$ associated to a collection of points p_i is defined by:
$\mathcal{V}_i = \{x \in \mathbb{R}^n : d(x,p_i) \leq d(x,p_r), \forall i \neq r\}$, where $d(x,y)$ denotes the distance between the points x and y.

Minkowski's addition of two sets X and Y is defined by:

$$X \oplus Y = \Big\{x + y : x \in X,\ y \in Y\Big\}.$$

Let $x(k+1|k)$ denote the value of x at time instant $k+1$, predicted upon the information available at time $k \in \mathbb{N}$. It is written $R \succ 0$ to denote that R is a positive definite matrix and $Q \succeq 0$ a positive semidefinite matrix. The length of the prediction horizon is denoted as N_p and the time step within prediction horizon is denoted by l.

The set of agents is represented by N_a and $\mathcal{I} \triangleq \{1,\ldots,N_a\}$ defines the collection of all agents indices.

The set of fixed obstacles is denoted by N_o and $\mathcal{I}_o \triangleq \{1,\ldots,N_o\}$ defines the collection of all obstacles indices.

The target/reference values for x is denoted by x^{ref} and the target/reference for subsystem i is denoted $x^{ref,i}$.

The spectrum of a matrix $M \in \mathbb{R}^{n\times n}$ is the set of its eigenvalues, denoted by $\Lambda(M) = \{\lambda_i : i =,\ldots,n\}$. A point x_e is a fixed point of a function f if and only if $f(x_e) = x_e$ (i.e., a point identical to its own image).

Acronyms

BMI - Bilinear Matrix Inequalities

DOF - Degree Of Freedom

IMC - Inter-Module Communication

LMI - Linear Matrix Inequalities

LP - Linear Programming

LQ - Linear Quadratic

LTI - Linear Time Invariant

MIP - Mixed-Integer Programming

MPC - Model Predictive Control

mRPI - minimal Robust Positive Invariance

NP-hard - Non-Polynomial hard

PWA - Piecewise Affine

QP - Quadratic Programming

RPI - Robust Positive Invariance

TP - Transportation Problem

UAV - Unmanned Aerial Vehicle

Chapter 1

Introduction

NOWADAYS, *multi-agent systems* have been identified as a new paradigm for a wide array of applications in control including, among others, scheduling and planning [Colombo et al., 2006], [Zhang et al., 2006], networked control [Dimeas and Hatziargyriou, 2005], hybrid control [Khosla and Dillon, 1997], [Fierro et al., 2001], condition monitoring [McArthur et al., 2007], automation [Buse et al., 2003], traffic and transportation networks [Burmeister et al., 1997], [Negenborn et al., 2008], water distribution networks [Overloop et al., 2010]. The multi-agent systems can be interpreted as a comparably recent concept with its origin in artificial intelligence, a branch of computer science. Following their transition in history, in the mid to late 1990's, multi-agent systems have replaced single agents as the computing paradigm in artificial intelligence [Wooldridge et al., 1995]. In the last twenty years, such systems started to be widely applied in different areas and for various purposes [Shen et al., 2006], from internet software to networked power stations, from the study of macro and micro biological interactions to complex physical phenomena and, most of all, allowing to industry to gain experience in the use of multi-agent systems technology and to evaluate its effectiveness [Marik and McFarlane, 2005]. Another important reason for their current ubiquity is that the complexity of the systems to be controlled is increasing more and more. It is often the case that a system is distributed in space, as well as in time and its treatment as a unique entity is not feasible anymore. Hence, the motivation for the increasing interest in multi-agent systems research is represented by their ability to enhance performance along the dimensions of computational efficiency, reliability, extensibility, robustness, maintainability, flexibility and reuse.

First of all, in order to find the benefits of multi-agent systems in control engineering, it is necessary to appropriately define them. There are various definitions of an *agent*, the foremost reason for this is due to the universality of the word and the fact that it cannot be owned by a single community. Most of the definitions come from computer science community and they all share three basic concepts, the notion of *agent*, its *environment*

and its *autonomy*. The definition of [Wooldridge et al., 1997] is often cited: *"an agent is a hardware or a software entity that is situated in some environment, and that is capable of autonomous action in the environment in order to meet its design objectives"*. As the definition indicates, the agents are either physical (hardware) entities, which is the case of robots, vehicles or pedestrians, in a physical environment (e.g., the control system) or virtual entities in a computing environment (e.g., data sources, computing resources), which is the case of software agents. However, for a discussion of agents in control, only the first case is of interest here, since there are direct applications in control engineering. [Wooldridge et al., 1997] also cites the first example of an agent as "any control system" using the thermostat device as an instance of a control system. It is situated in its environment, it reacts to temperature changes of environment and also exhibits a degree of autonomy.

A multi-agent system can be defined as a three-tuple comprising of a *set of agents* (homogeneous or heterogeneous), an *environment* and the *ability* to negotiate and interact in a *cooperative* manner [Wooldridge, 2002]. According to this definition we need to differentiate from the cases when we have several agents but the behavior is hostile (and not cooperative). Such a situation is to be found in adversary situations (e.g., a anti-missile trying to catch with a missile) or zero-sum games [Basar and Olsder, 1995] where the gain of one agent comes at the detriment of another agent. In other words, multi-agent systems represent a network of individual agents that share knowledge and communicate with each other in order to solve a problem that is beyond the scope of a single agent [Srinivasan, 2010]. In a multi-agent system, the action of an agent not only modifies its own environment but also the environment of its neighbors. This necessitates that each agent must predict the action of the other agents or receive meaningful information from them in order to compute its future control actions.

As the multi-agent systems community has established itself as an important area of computer science, there is much to be learned by seeking connections to control engineering. Nevertheless, the growing awareness of multi-agent systems in the research community rises in the same time basic questions related to their role in control engineering:

- are there dynamical systems which can be handled naturally in the multi-agent framework? In what circumstances do this make sense?
- are there specific advantages of using these constructions ? what strategies/ideas can be adapted from the computer science field?
- how to design such a system and how to implement it in a control engineering framework?

The presence of these questions leads to a number of challenging issues that need to be taken into account. The next step is to identify details about these challenges in order to provide the appropriate design methodologies and implementation approaches which are currently available.

In the sequel, we will present the state of the art in multi-agent dynamical systems from a control theoretic point of view, the main sources of inspiration in this endeavor being the comprehensive bibliographical study [Shamma, 2007] and the monograph of [Qu, 2009].

1.1 Multi-agent dynamical systems from a control theoretic perspective

Innovations in control engineering are crucial for many of today's challenges, such as improving data communication, information, processing, creating a sustainable energy and water supply, preventing and treating diseases. As already mentioned, multi-agent systems are becoming a staple topic of control research. To enumerate just a few of the success stories we may mention: The work of the Caltech team who conceived, designed, build and optimized the autonomous vehicle called "Alice" at the 2005 DARPA Grand Challenge [Murray et al., 2005]. Alice used a highly networked control system architecture to provide high performance, autonomous driving in unknown environment [Cremean et al., 2006]. The exploration of Mars started in 2003 by robots "Spirit" and "Opportunity", where the robots did an excellent job providing a fascinating description of the characteristics of the Mars soil [Norris et al., 2005]. Nowadays, the NASA planetary robotic missions are very well developed succeeding to performed many actions under autonomous control and with some kind of intelligence embedded in the robotic system of "Curiosity", the rover which explores now the planet. Other examples include, the control of formation flying satellites, also a DARPA project [Alfriend, 2010], the PITVANT flight operations using different types of Unmanned Aerial Vehicles (UAVs) [Gonçalves et al., 2011], [Sousa et al., 2004] and many other examples.

It has became obvious that the study of dynamical systems is essential in the control systems area. In the control of dynamical systems, control decisions are expected to be derived and implemented over real time. Feedback is used extensively to cope with uncertainties about the system and its environment.

It is often the case that a group of agents can seek together to achieve a global objective, or they can individually have their own objectives to pursue. Therefore, the *coordination* of the agents is required, that is, they need to coordinate their actions for improving the global efficiency. This is foremost called *cooperative control* of multi-agent systems and it refers to a group of dynamical agents that share information or tasks in order to accomplish a common (maybe not singular) objective [Murphey and Pardalos, 2002]. Just to give one example of cooperative multi-agent systems consider unmanned aircrafts patrolling wooded areas for fire or executing search and rescue operations [Valavanis, 2007].

The issues faced when dealing with the cooperative control of multi-agent dynamical systems include:

motion planning which responds to the classical question "how to get there?";

formation control in order to position themselves in a fixed position relative to each other;

centralized, distributed or decentralized control which handles the inter-agent communication problem and the structure of the control law;

constraints handling which usually need to be integrated in the design control;

stability depending on the nature of the problem (the stability may be understood as formation stability, tracking within some bounds or simply that the agents remain in a bounded region);

robustness with respect to disturbances and perturbations which affect the system needs to be considered; multi-agent specific issues also have to be handled (e.g., moving obstacles, estimation of other agents' state).

1.1.1 Multi-agent dynamical systems motion planning and real-time control

The objective in the multi-agent systems motion planning problem is to generate real-time trajectories to guide a collection of agents to their destination while avoiding obstacles and also avoiding each other. The destinations can be specified as absolute or relative locations, a certain coverage diagram or a general direction of flow [Shamma, 2007]. The most common examples include *formation keeping* problem [Tillerson and How, 2002], *rendevouz* problem [Dimarogonas and Kyriakopoulos, 2007] or *swarm* of vehicles [Olfati-Saber, 2006]. Next, the different approaches used in the literature are presented.

Graph theoretic methods model the execution of a multi-agent system as a graph which contains high-level description of the system topology in terms of objects referred to as vertices and edges [Mesbahi and Egerstedt, 2010]. This approach employs connections between graphs and their algebraic representation in terms of adjacency, Laplacians and edge Laplacian matrices as well as different aspects of the spectrum of graph Laplacian [Blondel et al., 2009].

Game theory approaches are concerned with the interactions between multiple decision-makers, and there are clearly relevant to cooperative control [Marden and Shamma, 2007]. The convergence to the Nash equilibrium in zero sum or general sum games is of great interest in game theory. There is also a variety of important concepts, such as

mechanism design, bargaining, coalition theory, and correlated equilibrium. All these concepts are developed with the intention of being models of social phenomena. The adversarial interactions between the agents are often tackled using game theory approaches. In adversarial situation the agents must address the possibility of other adversary agents that are also capable of strategic planning [Bauso et al., 2008].

Potential field approaches consists in constructing a scalar function called the potential that has a minimum, when the agent is at the goal configuration, and a high value on obstacles. Everywhere else, the function is decreasing towards the goal configuration, so that the agent can reach the goal by following the negative gradient of the potential. A shortcoming of the method is the possibility of traps (local minima), a problem that can be overcome by using ideas such as a randomized potential field. In [Koditschek, 1992] a historical review of the potential field approach can be found.

Viability theory designs and develops mathematical and algorithmic methods for investigating the adaptation to viability constraints of complex systems that combine both continuous and discrete dynamics and both control inputs and disturbances [Aubin et al., 2011]. Such uncertain, hybrid systems systematically appear in the control of large scale multi-agent systems such as automated highways or air traffic management. In this context, problems as controlled invariance represent challenging issues which needs to be taken into account [Crück and Saint-Pierre, 2004]. For example, in an automated highway system the controller of each car should be design such that, without regard of the uncertainty about the actions of the other cars, collisions between cars are avoided [Gao et al., 2004].

Optimization-based control approaches in general terms cover the control design based on optimization techniques. These can cover the classical optimal control, the LMI-based techniques, model predictive control or interpolation-based techniques. From a methodological point of view, a particularly useful approach of control for collaborative systems is receding horizon optimization-based control, also known as *Model Predictive Control* (MPC) [Goodwin et al., 2005], [Rawlings and Mayne, 2009]. An important aspect of the predictive control approach is its ability to handle generic state and control constraints. This issue will also be the main focus of the present thesis.

The last class of methods deserves a special attention. It is known that MPC is not a new design approach and it has extensively been applied to systems with slow dynamics (e.g., chemical process plants), allowing the use of a sampling rate large enough with respect to the optimal control computations between samples [Murray, 2009]. Moreover, these systems are governed by constraints on states, inputs or a combination of both. In recent years, it has become possible to apply receding horizon strategy to systems with faster dynamics, due to cheap computational power and theoretical advances on the optimization algorithms and their specific tailoring for the structure of MPC problems [Bock et al., 2007], [Diehl et al., 2009]. There is a broad variety of practical applications that can be treated by receding horizon control in process control and related industrial applications [Qin and Badgwell, 2003], [Morari et al., 2002], road traffic networks [Hegyi

et al., 2005], [Baskar et al., 2006], water control for irrigation canals, water supply and sewer networks [Negenborn et al., 2009], [Ocampo-Martinez and Puig, 2009], [Overloop et al., 2010]. Also, this technique is gaining an increasing attention in other fields as mine planning [Goodwin et al., 2008], [Goodwin et al., 2006], food processing [Shaikh and Prabhu, 2007], drug delivery [Parker et al., 2001], [Bleris et al., 2007] to cite only a few. Analyzing this broad field of application, the natural conclusion is that, the receding horizon control is, by its optimization-based approach a natural candidate for multi-agent problems, in that such a technique can admit very general objectives, as for example, convergence to a formation, trajectory tracking, while taking into account collision avoidance constraints.

1.1.2 Multi-agent formation description and management

The characterization and the convergence towards a formation represents classical issues for the control of cooperative systems. Research on formation control dealt with ground vehicles [Fax and Murray, 2004], [Lin et al., 2005], surface and underwater autonomous vehicles (AUVs) [Fossen, 1994], [Aguiar et al., 2006], unmanned aerial vehicles (UAVs) [Scharf et al., 2004], [Schouwenaars et al., 2005], micro and macro satellite clusters [Schaub et al., 2000]. The definition of formation is specified in many different ways in the literature, as for example, the rigid formation [Saber et al., 2003], [Fontes et al., 2009] or a desired figure in the plane [Balch and Arkin, 1998]. According to [Qu, 2009], a formation represents an organization of a group of agents moving together under certain constraints, in a confined space, with common goals. Control to formation is in a way a *consensus*[1] problem, since in order to converge to formation the vehicles have to achieve, among other things, the same velocity, which is called *flocking motion* [2] in the literature.

Several important elements are needed for solving the formation control problem:

- how to characterize the formation?
- what are the control strategies used for formation control?
- how to guarantee the convergence to a predefined formation?
- how to preserve the formation?

Usually the formation is specified through desired inter-agent distances (e.g., diamond, wedge, line, triangle) in the static case and then, in real-time the specified distances

[1] Consensus problem is being understood as having the agents brought to a global agreement on a state value [Shamma, 2007].

[2] Flocking motion is being understood technically as a convergence property on the agent velocity and their relative distances, while a mission will be the convergence of the flock to a specified point in the workspace [Shamma, 2007].

need to be maintained while the agents are in motion. There are also cases where a *scale invariant formation* is necessary [Mesbahi and Egerstedt, 2010]. This type of formation specification makes sense in applications where the environment is cluttered and a scaled contraction or expansion of the formation may be required to negotiate the environment[3]. An alternative approach to this type of formation specification is given for example in [Fontes et al., 2009], where a switch among a collection of a priori predefined formations is employed. Using graph theory approaches, examples of various ways by which formations can be specified include deviations from desired positions, as is the case in the work of [Ogren et al., 2001], deviations from desired inter-agent distances [Jadbabaie et al., 2003], or as dissimilarities between graphs encoding the desired and actual formations [Ji and Egerstedt, 2007].

In a more general framework, the formation control problems can be defined first by a desired geometric shape and second, as an *assignment* problem [Mesbahi and Egerstedt, 2010]. Once, a set of agents is to achieve a predefined configuration, one has to decide if the identity of each individual agent has a particular importance (i.e., the indices in the predefined target configuration need to correspond to particular indices associated with the actual agents). If it is the case, then an assignment problem needs to be integrated in the formation control design in order to decide which agent should take on what role in the formation. It is well known that the standard *static assignment problem* is a special case of the so-called Hitchcock Transportation Problem (TP) [Palekar et al., 1990], [Burkard, 2002]. Furthermore, the Transportation Problem is a *combinatorial optimization problem*, with a potentially prohibitive computational requirement for large formation. In the case of linear assignment problem, the optimal assignment can in fact be efficiently computed. One classical method to do this is the so-called "Hungarian algorithm" [Kuhn, 1955], [Hallefjord Kurt, 1993] with a polynomial complexity of $\mathcal{O}(n^3)$. In the *dynamic assignment problem*, agent locations, target locations, and target constraint sets are all time-varying, which induces time-variations in the overall utility function. The problem is then to design dynamic agent-target assignment policies whose performance is measured by time-aggregated behavior such as the rate of successful target assignments. The assignment problem formulation has received much attention in cooperative control studies therefore many algorithms of polynomial complexity exists to solve it [Bertsekas, 1998], [Bertsekas and Castanon, 1991], [Bertsekas, 1992]. In [Schumacher et al., 2002], multiple assignment problems with evolving subsets of the agents and tasks are iteratively solved, thus allowing their algorithm to produce a multiple task tour solution that simulations show to be close to optimal for their scenario. The work uses a centralized approach in which each agent solves the same problem independently and then implements its own part of the resulting assignment. It is therefore highly dependent on the agents starting from the same information in order to guarantee that they arrive at the same solution.

[3]Consider the example of three agents which are to traverse a narrow passage while maintaining a desired shape. The way this can be achieved is by scaling the formation through the maneuver so that a tighter formation is used while going through the passage.

One of the most used approaches to formation control is the so-called *leader-follower* [Das et al., 2002], [Mariottini et al., 2009] where one agent designed as the leader provides a route towards the destination for the whole group. The other agents follow the leader and regard it as the reference point while maintaining a desired distance and orientation to the leader. Due to the very important role of the leader, any failure on its part will affect the whole system. Nevertheless, the leader-follower structure is easy to implement since a reference trajectory is clearly defined by the leader and the internal formation stability is induced by the stability of the individual vehicles control laws [Consolini et al., 2009]. A practical framework for on-line planning and control of multiple mobile robots moving in a leader-follower formation in a dynamically changing environment was proposed in [Hao, 2004]. The authors develop trajectory plans for the leader and use a graph search method to find a collision free and deadlocks free paths quickly for the formation group. There is also *virtual-structure* formation approach where the entire formation is treated as a single rigid body and the agents maintain a specific geometry among each other [Tan and Lewis, 1996], [Ren and Beard, 2004], [Beard et al., 2001]. For example, in [Tan and Lewis, 1996], a strategy is presented to arrange a large-scale homogeneous group of agents in a geometric formation by using potential functions with preset attachment sites. However, there is a limited number of formations that can be achieved and the method is computationally expensive.

Regarding the stability analysis, the Lyapunov method is extensively used to study the asymptotic behavior of multiple agent formation. [Jadbabaie et al., 2003], [Tanner et al., 2005] investigate the motions of vehicles modeled as double integrators. Their goal is for the vehicles to achieve a common velocity while avoiding collision. The control laws involve graph Laplacians for an associated undirected graph but also nonlinear terms resulting from artificial potential functions. Rather than reaching a predetermined formation, the vehicles converge to an equilibrium formation that minimizes all individuals potentials. Furthermore, [Tanner et al., 2007] prove that the stability of motion and convergence to the destination is not affected by the control discontinuities induced by a dynamically changing interconnection topology, and are being established using results from algebraic graph theory and Lyapunov stability analysis for nonsmooth dynamical systems.

1.1.3 Constraints handling

In many multi-agent control engineering problems, collision avoidance represents a fundamental issue that needs to be integrated in the design strategy. Usually, these issues are addressed as *soft* or *hard constraints*. In the first case, usually are used Potential Filed approaches, where the constraints are penalties in the cost function, and the second case, refers to constrained optimization problems where the constraints appear explicitly. The choices are dual, the first approach considers a more complicated cost function and simple (or no-) constraints and the second approach considers a simple cost function but

relatively complicated constraints. Actually, the big problem comes not from the existence of the constraints but from their nature. Imposing collision and obstacle avoidance gives rise to non-convex constraints. This means in turn, a non-convex and (possibly) a non-connected feasible region, which is known to loose some important computational advantages of the established (optimization or design) techniques, which are effective under some appropriate convexity assumptions [Boyd and Vandenberghe, 2004].

In the literature, various control methods for solving the collision avoidance problem are related to the potential field approach [Tanner et al., 2007], graph theory [Lafferriere et al., 2004] or other optimization-based approaches which handle indirectly the constraints by penalty terms in the cost function. One shortcoming of all these methods is that the constraints may not be satisfied. A common point of most publications dealing with the collision avoidance problem is the hypothesis of "*punctiform*" agents, which is far from the conditions in real world applications. In many of them, the relative positioning between agents becomes important, such as large interferometer construction from multiple telescopes [Schneider, 2009] or the air traffic management, two aircraft are not allowed to approach each other closer than a specific alert distance [Dimarogonas et al., 2006]. Spherical shape for the agents is considered in [Dimarogonas and Kyriakopoulos, 2006] and a navigation function is used in order to guarantee collision free behavior in the group formation, [Barnes et al., 2009] consider ellipsoid safety regions for the agents moving in a potential field environment. Other works, as for example, [Patel and Goulart, 2011] propose a gradient-based optimization algorithm for trajectory generation for aircraft avoidance maneuvers where the obstacles are defined as convex regions. Nevertheless, it is important to mention that none of these works do not consider obstacles or safety regions which cover the natural equilibrium position zero (for example the origin for linear time invariant dynamics).

Methods based on *Mixed-Integer Programming* (MIP) (see the comprehensive monography [Jünger et al., 2009]) have the ability to include *explicitly* non-convex constraints and discrete decisions in the optimization problem. There is a growing literature about optimization problems, which can be formulated through the use of MIP techniques. For example, [Schouwenaars et al., 2001], [Richards and How, 2005], [Ousingsawat and Campbell, 2004] focused their work on optimization of agent trajectories moving by the obstacles. Multi-vehicle target assignment and intercept problems are studied by [Earl and D'Andrea, 2001], [Beard et al., 2002]. MIP, was also useful to coordinating the efficient interaction of multiple agents in scenarios with many sequential tasks and tight timing constraints (see, [Richards et al., 2002], [Schumacher et al., 2003]). In Bemporad and Morari [1999], [Bemporad et al., 2000], the authors used a combination of MIP and Model Predictive Control (MPC) to stabilize general hybrid systems around equilibrium points. [Bellingham et al., 2002] introduced MIP in a predictive control framework to plan short trajectories around nearby obstacles.

However, despite its modeling capabilities and the availability of versatile solvers, MIP-based approach has some serious numerical drawbacks. As stated in [Garey and Johnson,

1979], mixed-integer techniques are classified as being NP-hard, i.e. the computational complexity increases exponentially with the number of binary variables used in the problem formulation. Consequently, these methods could not be fast enough for real-time control of systems with large problem formulations. There has been a number of attempts in the literature to reduce the computational requirements of MIP formulations in order to make them attractive for real-time applications. In [Earl and D'Andrea, 2005], an iterative method for including the obstacles in the best path generation is provided. Other references, like [Vitus et al., 2008], consider a predefined path constrained by a sequence of convex sets. In all of these papers, the binary variables reduction is not tackled at the MIP level, but, instead, the original decision problems are reformulated in a simplified MIP form.

1.1.4 Centralized vs. distributed vs. decentralized control

From the point of view of inter-agent communication we can differentiate between several strategies [Scattolini, 2009]. They are on one hand determined by numerical difficulties (too hard to solve a certain optimization) or by communication limits/geographical distribution. *Centralized control* is, from a theoretical point of view, the easiest way to handle such problem, since it assimilates the multi-agent system with a single extended system. Needless to say this increases the computational effort and make some possibly unrealistic assumptions about communication range and capacity. The other extreme is represented by the *decentralized control* where the communication and inter-dependence in general is reduced to a minimum. In between, and maybe the most challenging case is the *distributed control* which aims for the optimality of the centralized approach with the costs of the decentralized one. More precisely, the distributed control usually means a decomposition of a large scale system into a set of several smaller subsystems ("neighborhoods"). The rationale of this approach is to provide subsystems which have fewer decision variables and are affected by fewer constraints (thus making the optimization problems easier to solve). This design requirement means that we aim for subsystems which are loosely inter-coupled, i.e., a given subsystem is affected by only a few other subsystems [Scattolini, 2009].

As a field of application, the design of vehicle formation through the use of centralized MPC is detailed in [Dunbar and Murray, 2002], where the authors formulate and numerically solve a nonlinear, constrained MPC problem, where the terminal state consists of a family of equilibria. In their approach there is no leader/follower architecture, although a formation reference is defined and a virtual leader could be considered. The control objective is to steer the set of states to an equilibrium formation, therefore the vehicles are stabilized to an asymptotically stable invariant terminal set, rather than a precise location for each vehicle in the formation. In [Keviczky et al., 2006] and [Dunbar and Murray, 2006], decentralized and distributed MPC algorithms are developed for systems with nonlinear, decoupled dynamics and the focus is on the synthesis of a control law, not the solution of optimization problems. The authors present first the

centralized problem and then, for the decentralized and distributed problems break the centralized one into distinct MPCs of smaller size. Next, [Li et al., 2005] presents an algorithm for distributed MPC of linear systems without constraints, where the sub-problems are solved in parallel until the Nash equilibrium is reached. Furthermore, [Venkat et al., 2005] reports a cooperative-based distributed MPC algorithm for linear systems that converges to a centralized solution and gives appropriate conditions for closed-loop stability. Similar results on optimization are obtained in [Camponogara and De Oliveira, 2009], where a decomposition of the global problem is consider in order to ensure cooperative behavior of the agents. Furthermore, [Negenborn et al., 2008] addresses the solution of optimization problems within a distributed predictive control framework, where Lagrangian duality is applied in order to handle coupling variables among neighboring agents. Finally, other works consider the decentralized navigation using gradient-based methods [Dimarogonas et al., 2006], [Tanner et al., 2007], [Barnes et al., 2009].

1.2 Thesis orientation

From the literature review presented above it is clear that the multi-agent domain is extremely large and heterogeneous. Thus, in this thesis we will limit to a certain area an explore what can be done under certain limitations and assumptions.

The main idea of the present work is to use optimization-based decision making and the set-theoretic tools and apply them to all aspects of the multi-agent problem. This is of course, not entirely new but here we employ these principles in a systematic manner. In this context is normal to have some limiting assumptions. For example, we consider bounded perturbations, disturbances and sometimes, obstacles. Without this basic assumptions is not possible to have bounded set representation and thus, part of the developments cannot reach final stability/boundedness conditions. Still, such assumptions are not excessively harsh since, in practice, it is natural to have bounds for noises and perturbations, and even for the neighborhood on which a specific system is able to deploy sensing capacities.

Since we use sets in the problem (and specifically model description) it is also natural to consider hard constraints. This means that, usually, the optimization problems under consideration are constrained. The alternative will be to consider them implicitly via some appropriate penalty functions - as they are e.g., in the Potential Field methods. Still, they can be adapted and we have shown this in the manuscript. The set-theoretic framework allows the analysis of fundamental properties as the invariance of a set for some dynamics. Using this notion, it will be possible to describe safety regions around each agent and these regions will not be recomputed at each instant of time because they are time invariant.

We are also interested characterizing formations from a geometrical point of view. The issues of existence, stability and uniqueness of a formation can be rephrased as the existence, stability an uniqueness of a fixed point in some appropriate extended space. In the presence of non-convex constraints (as imposed for example, by obstacle and collision avoidance), the fixed point becomes restricted by adversary constraints making thus the natural equilibrium infeasible. We consider this analysis interesting both, as a theoretical result and because of the links it provides with set theory and limit behavior of an agent.

1.3 Contributions of the thesis

This thesis can be placed in a line of research containing the results [Richards and How, 2002], [Richards and How, 2005], [Tanner et al., 2007]. More precisely, we are focusing on a multi-agent systems optimization-based control scheme based on the combination of Model Predictive Control (MPC) and set theoretic methods. With respect to these studies, we enhance the construction methods (with contributions towards the geometrical interpretation) and open new directions in the exploration of formation control problem of multiple agents in the presence of non-convex constraints. In particular, we put the formation design and the collision avoidance conditions as geometrical problems and proceed to solve them into a mixed integer programming context. Of course, this is not entirely new but we try in this manuscript to look at the geometrical meaning of the problem and to optimize in regards to it (from complexity and performance points of view). This will allow reducing the computational cost and the efficient handling of an increased number of constraints.

Not in the least, we give a significant amount of attention to the generation and convergence towards a formation. Moreover, we proceed to provide guarantees of uniqueness and existence of such a formation as conditions on the eigenstructure of the state matrices of the agents. These properties are dependent on the optimization scheme to be considered, but at least, in the centralized (and, in certain conditions, distributed) setting, they can be guaranteed.

It is worth mentioning that all these constructions have a practical justification. There are situations where a tight/predefined formation is required [Sousa et al., 2000], [Girard et al., 2001], [Girard et al., 2005] and where our results can be applied.

In [Prodan et al., 2011b] we decomposed the formation control problem in two separate problems. First, the ideal configuration was described starting from the definitions of ultimate bounds for the agents dynamics in the presence of additive disturbances. This choice is adopted in order to guarantee a degree of robustness despite the fact that the real-time control is performed using nominal prediction models. Once the positions in the formation are established, safety regions are defined around the nodes of the formation.

Second, in real-time, a receding horizon combined with task assignment relative to the minimal configuration is employed.

Furthermore, in [Prodan et al., 2012c], we brought enhancements in the previously described control design method, which enables the stabilization of the multi-agent formation. We show that, for the convergence to the predefined formation an additional fixed point constraint (i.e., the target positions are also equilibrium points for the considered dynamics) must be taken into account. Moreover, we have obtained considerable advances in this direction by using efficiently Mixed-Integer Programming (MIP) techniques in order to derive a coherent description of the feasible region in the solution space. A novel method of reducing the number of binary variables for representing the non-convex feasible region was first presented in [Stoican et al., 2011b] and then extended to non-convex non-connected feasible regions in [Stoican et al., 2011c]. The notable improvements which enabled the minimization of the number of constraints were all gathered in the paper [Prodan et al., 2012e].

We have explored the formation control problem from a centralized [Prodan et al., 2011b], [Prodan et al., 2012c], distributed [Prodan et al., 2012f] and decentralized [Prodan et al., 2012d] point of view. In each of these papers, we highlighted the strengths and weaknesses of theses approaches. The centralized MPC implementation offers the best theoretical guarantees for the multi-agent formation stability but is numerically cumbersome. On the other hand, we partitioned the set of agents into neighborhoods and solve a distributed control problem [Prodan et al., 2012f].

Since the MIP formulation is inherently difficult to solve, in [Prodan et al., 2012d] we discuss a decentralized approach for the formation control problem using a Potential Field methodology [Tanner et al., 2007]. In this case, the constraints are no longer hard but rather soft in the sense that we impose penalties in the cost function. A particular contribution here was represented by the use of polyhedral norm in order to construct penalty functions that take into account the shape of the safety region associated to each agent.

As previously detailed, there are various methods for formulating and solving the multi-agent optimization problem. The common factor of all these methods is the presence of non-convex constraints. Hence, solving problems over non-convex regions does not represent a new issue in the literature, but in [Prodan et al., 2011a] and [Prodan et al., 2012b] we go further, in the sense that the natural unconstrained equilibrium point (the origin for the case of linear dynamic systems) becomes infeasible. More precisely, the type of constraints that we consider makes the convergence of an agent dynamics towards the origin impossible to fulfill. In [Prodan et al., 2011a] we propose an explicit solution to this problem for second-order dynamical systems and extend [Prodan et al., 2012b] by using a dual-mode control approach.

The theoretical methods detailed above were tested on several benchmarks, simulations and practical examples. In [Prodan et al., 2012a], we presented software-in-the-loop

simulations for the predictive control for trajectory tracking of Unmanned Aerial Vehicles (UAVs). In order to generate a feasible trajectory we made use of the differential flatness concepts [Fliess et al., 1995]. The proposed trajectory tracking mechanism takes into account way-point conditions and also allows to obtain off-line linearizations by the use of a precomputed Voronoi diagram of the linearized models along the flat trajectory. Furthermore, the predictive control approach was validated through real flight test results for the control of an autonomous UAV.

In [Prodan et al., 2010] we solve the trajectory tracking problem for multi-agent formation under collision avoidance constraints. Besides maintaing a safety distance between the agents, we were also interested in imposing velocity constraints for an agent. These represent, for example, safety limits, such as a minimum maneuvering velocity near an obstacle or another agent. Based on the information received from the MPC formulation, the constraints are taken into account, leading the agents to follow the reference trajectory in a formation which depends on the geometry of the non-convex constraints.

We provide in the sequel the complete list of publications accepted/submitted to various conferences and journals:

Accepted journal papers:
• **I. Prodan**, F. Stoican, S. Olaru, S.-I. Niculescu: Enhancements on the Hyperplanes Arrangements in Mixed-Integer Programming, *Journal of Optimization Theory and Applications*, Vol. 154, No.2, pp. 549-572, DOI 10.1007/s10957-012-0022-9, August 2012.
• **I. Prodan**, S. Olaru, C. Stoica, S.-I. Niculescu: Path following with collision avoidance and velocity constraints for multi-agent group formations. *Annals of the University of Craiova, Series: Automation, Computers, Electronics and Mechatronics*, ISSN: 1841-0626, volume 7(34) no.2, pp. 33-38, 2010, http://www.ace.ucv.ro/analele/content2010vol7nr2.html (a short version was presented at Sinaia, Romania).
• **I. Prodan**, S. Olaru, C. Stoica, S.-I. Niculescu: Predictive control for trajectory tracking and decentralized navigation of Multi-Agent formations. *International Journal of Applied Mathematics and Computer Science*, accepted, to appear in 2012.
Submitted journal papers:
• **I. Prodan**, S. Olaru, R. Bencatel, J.B. Sousa, C. Stoica, S.-I. Niculescu: Predictive Control for Autonomous Aerial Vehicles Trajectory Tracking. *Control Engineering Practice*, 2012 (second round review).
Book chapters:
• **I. Prodan**, S. Olaru, C. Stoica, S.-I. Niculescu: On the Tight Formation for Multi-Agent Dynamical Systems. *Agents and Multi-agent Systems - Technologies and Applications*, Vol. LNAI 7327, pp. 554-565, DOI 10.1007/978-3-642-30947-2, Springer, 2012.
• **I. Prodan**, F. Stoican, S. Olaru, C. Stoica, S.-I. Niculescu: Mixed-Integer Programming Techniques in Distributed MPC Problems, *Distributed MPC Made Easy*, Springer, accepted, to appear in 2013, http://distributedmpc.net/.

Accepted conference papers:

• **I. Prodan**, R. Bencatel, S. Olaru, J.B. Sousa, C. Stoica, S.-I. Niculescu: Predictive Control for Autonomous Aerial Vehicles Trajectory Tracking. *The IFAC Nonlinear Model Predictive Control Conference*, IFAC NMPC'11, Noordwijkerhout, The Netherlands, 23-27 August 2012.

• **I. Prodan**, G. Bitsoris, S. Olaru, C. Stoica, S-I. Niculescu: On the Limit Behavior for Multi-Agent Dynamical Systems. *The IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles*, NCUV'12, Porto, Portugal, 10-12 April 2012.

• **I. Prodan**, S. Olaru, C. Stoica, S.-I. Niculescu: Predictive control for trajectory tracking and decentralized navigation of Multi-Agent formations. *The 4th International Conference on Agents and Artificial Intelligence*, ICAART'12, Vilamoura, Portugal, 6-8 February 2012.

• **I. Prodan**, S. Olaru, C. Stoica, S.-I. Niculescu: Predictive control for tight group formation of Multi-Agent Systems. *The 18th World Congress of the International Federation of Automatic Control*, IFAC'11, Milan, Italy, 28 August-2 September 2011.

• **I. Prodan**, S. Olaru, C. Stoica, S.-I. Niculescu: On the limit behavior of Multi-agent Systems. *The 8th International Conference on Informatics in Control*, Automation and Robotics, ICINCO'11, Noordwijkerhout, The Netherlands, 28-31 July 2011.

• F. Stoican, **I. Prodan**, S. Olaru: On the hyperplanes arrangements in Mixed-integer techniques. *The 30th American Control Conference*, ACC'11, California, USA, 29 June -1 July 2011.

• F. Stoican, **I. Prodan**, S. Olaru: Enhancements on the hyperplanes arrangements in Mixed-Integer techniques. *The 50th IEEE Conference on Decision and Control and European Control Conference*, CDC-ECC'11, Orlando, Florida, USA, 12-15 December 2011.

• **I. Prodan**, S. Olaru, C. Stoica, S.-I. Niculescu: Collision avoidance and path following for multi-agent dynamical systems. *The 9th International Conference on Control, Automation and Systems*, ICCAS'10, Gyeonggi-do, Seoul, Korea, 27-30 October 2010.

• **I. Prodan**, S. Olaru, C. Stoica, S.-I. Niculescu: Path following with collision avoidance and velocity constraints for multi-agent group formations. *The 14th International Conference on System Theory, Control and Computing*, ICSTCC'10, Sinaia, Romania, 17-19 October 2010.

• **I. Prodan**, G. Bitsoris, S. Olaru, C. Stoica, S.-I. Niculescu: Dual-Mode Constrained Control for Dynamical Systems with Geometric Adversary Constraints. *The 5th IFAC Symposium System Structure and Control*, IFAC SSSC'13, Grenoble, France, 4-6 February 2013.

Submitted conference papers:

• **I. Prodan**, S. Olaru, F. Fontes, C. Stoica, S.-I. Niculescu: From trajectory tracking to path following in view of optimization-based control for autonomous aerial vehicles, *The 13th IEEE European Control Conference*, ECC'13, Zurich, Switzerland, 17-19 July 2013.

• F. Stoican, **I. Prodan**, S. Olaru: Further Enhancements on the Hyperplane Arrangements in Mixed-Integer Programming Techniques. *The 13th IEEE European Control*

Conference, ECC'13, Zurich, Switzerland, 17-19 July 2013 (invited session).

1.4 Organization of the manuscript

This thesis (excluding the present chapter) is partitioned into six chapters:

Chapter 2 presents the optimization-based control approach that we adopt in order to deal with the control of multi-agent systems in the presence of constraints. Firstly, a short overview of receding horizon control and its capability to handle state and control constraints is provided. Secondly, the main ingredients needed in the multi-agent predictive control context are motivated and described one by one. Differential flatness concepts play an important role in the resolution of a trajectory planning problem and it occupies an important part of the presentation. Furthermore, for a robust approach, the use of set-theoretic methods is invoked. Finally, the last part of the chapter, presents an important contribution towards an efficient geometric description of non-convex non-connected feasible regions through the use of Mixed-Integer Programming (MIP).

Chapter 3 concentrates on the optimization-based control of multiple agents having independent dynamics while achieving a global objective, such as a tight formation with desired specifications and collision free behavior. Invariant safety regions around the agents are constructed in order to compensate the effects of the disturbances affecting their dynamics. The formation control problem is decomposed in two separate problems. "Off-line", a minimal configuration is constructed taking into account the non-convex constraints imposed by the safety regions. In "real-time", a receding horizon optimization combined with task assignment relative to the minimal configuration is employed. MIP techniques are efficiently used here in order to achieve a coherent description of the feasible region in the state space. Furthermore, the proposed formation control approach is presented from centralized, distributed and decentralized points of view. In the last part of the chapter, a potential field approach is employed.

Chapter 4 presents a detailed analysis of the limit behavior for a linear dynamical system in the presence of geometric adversary constraints. This type of constraints makes the convergence of an agent dynamics towards the origin impossible to fulfill. Therefore, first, the fixed points and the invariance properties for the system state trajectory are described while avoiding a convex region containing the origin in its strict interior. Second, the attention is focused on ensuring the stability over the feasible region of the state-space using a dual-mode strategy. Finally, the last part of the chapter extends this analysis to multi-agent formations.

Chapter 5 provides in a first section, software-in-the-loop simulations and real-time flight tests results for the predictive control of Unmanned Aerial Vehicles (UAVs).

A specified trajectory is generated by taking into account way-point conditions and furthermore, allowing the use of off-line linearizations of the nonlinear vehicle model. Discussions, concluding remarks and improvements directions on the results obtained are presented in the last part of the section. Furthermore, in a second section, simulations results for trajectory tracking of multi-agent formation are presented.

Chapter 6 completes the thesis with conclusions and discussions of future directions.

Pictorially, one can view the previously described chapters as shown in Figure 1.1, where the edges suggest dependencies between the various chapters.

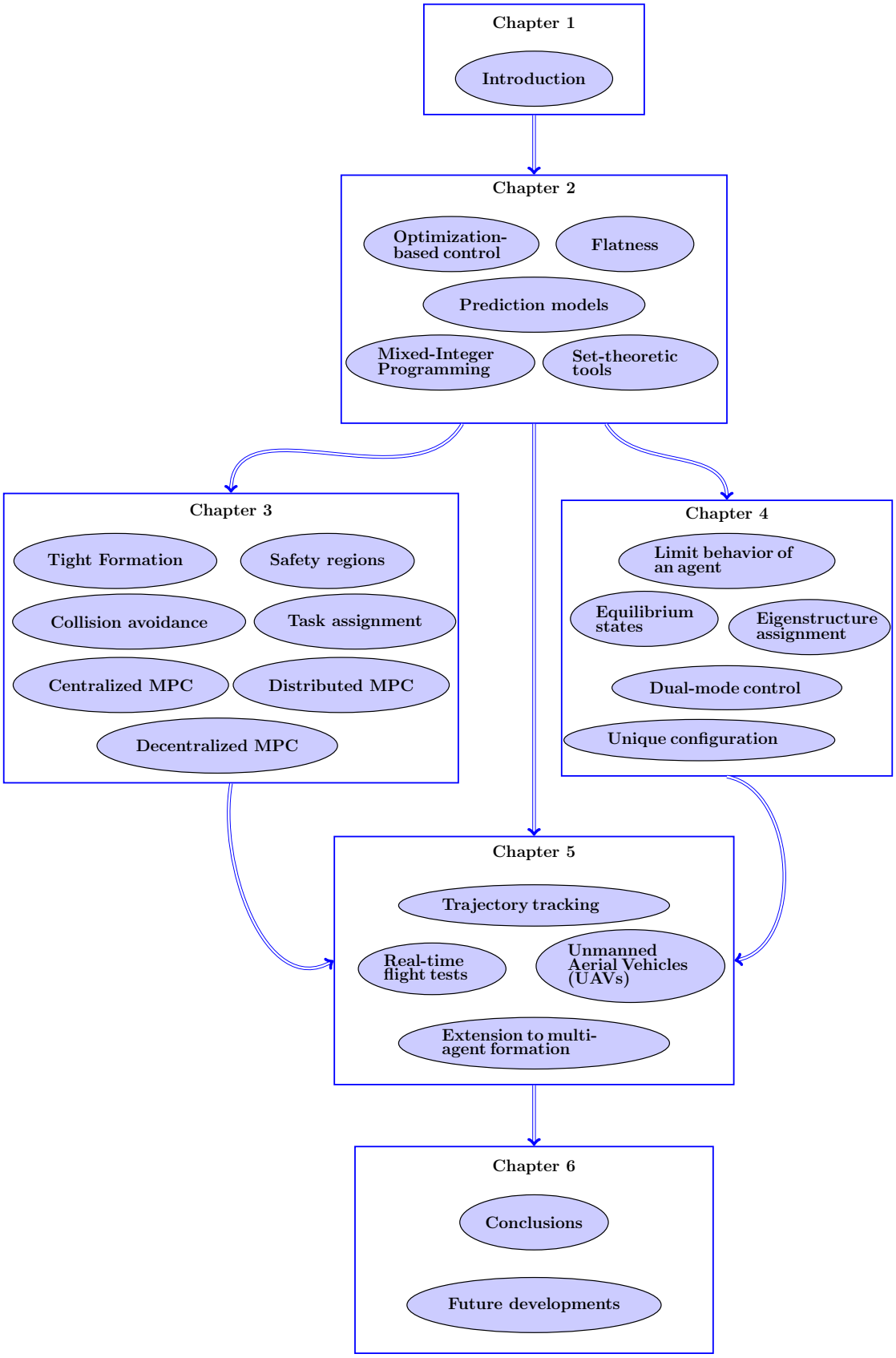


FIGURE 1.1: Dependencies between the thesis chapters.

Chapter 2

An optimization-based approach for control of cooperative systems

The present chapter builds on the foundation of *optimization-based control* and gradually paves the way towards the integration of *set-theoretic concepts* in the context of *cooperative systems*. Moreover, several particularities of the constrained control of Multi-Agent Dynamical Systems will be carefully examined and discussed. From a methodological point of view, a particularly useful approach of control for collaborative systems is the so-called receding horizon control, also known as *Model Predictive Control* (MPC) [Goodwin et al., 2005], [Rawlings and Mayne, 2009]. An important aspect of predictive control approach is represented by the ability to handle generic state and control constraints and will represent the mainstream of the present thesis. As detailed in the Introduction chapter, the receding horizon control is, by its optimization-based approach, a natural candidate for multi-agent problems. This technique can admit very general objectives, as for example, convergence to a formation, trajectory tracking, while taking into account collision avoidance constraints. In the first part of the present chapter we will come back to these statements in a formal manner by providing a brief overview of optimization-based control.

The dynamical systems under consideration, represented by their models for the prediction purposes, can be either continuous-time or discrete-time, either homogeneous or heterogeneous, either linear or non-linear, and constrained. In the present manuscript we work, for the most part, with simplified dynamics for the (sub-)systems generally inherited from vehicle dynamics, that is, those with double integrator-type of models, linear time-invariant models and unicycle nonlinear models, also in the presence of bounded disturbances and constraints. Note that we have chosen simplified dynamics and their transportation analogy in order to make the interpretation of the numerical results straightforward. However, the constraints handling, the control design and the resulting techniques hold for general formulations of dynamics and constraints.

For tasks involving cooperation among a group of agents, trajectory tracking is often crucial for achieving the cooperation objective. It is often the case that a *reference trajectory* has to be precomputed. When talking about references signals, it is important to point out the difference between two "close" notions: reference trajectory and reference path. The latter only provides a desired route for the agents for which it may not necessarily exist a feasible input. In this sense, the former appears more appealing, since it provides simultaneously both feasible input and state, the disadvantage being its time-dependence which often impose an additional constraint on the real-time functioning, while the reference path remains time-independent. Of course, computing feasible reference input and state signals is, generally, a difficult task. We choose to use in the present work one of the few generic tools, those based on *differential flatness* [Fliess et al., 1995], [Lévine, 2009] in constructing a reference trajectory.

If one accepts MPC as a design framework and have a defined reference trajectory available, it can basically design a nominal control loop, but robustness needs to be taken into account with appropriate analysis tools and notions. For a robust approach, the use of sets will be advocated in the present work. Whenever the noises and disturbances are bounded, we will pursue the construction of bounded invariant sets characterizing thus the system dynamics. For example, we can construct safety regions around agents in order to guarantee collision avoidance for any values of the bounded noises. Therefore, in the present chapter, a particular emphasis will be put on set-theoretic concepts [Schneider, 1993], [Aubin et al., 2011] in control and *invariance notions* [Blanchini and Miani, 2008]. For numerical reasons, the family of sets that we choose are the *polyhedral sets*. They strike an excellent balance between flexibility of representation and numerical implementation of the algorithms. More specifically, we will use them in the context of multi-agent systems for describing safety regions, obstacles and feasible regions. A related notion, the *polyhedral norm* will be also employed in order to construct a so-called polyhedral function and sum function, which will further represent the basis for the construction of "exclusion" functions (or penalty functions).

Collision avoidance is often the most difficult problem in the context of managing multiple agents, since certain (static or dynamic) constraints are non-convex. This happens because the evolution of a dynamical system in an environment presenting obstacles can be modeled only in terms of a non-convex feasible region, i.e., the agent state trajectory has to avoid a convex (union of convex) region(s) representing an obstacle (static constraints) or another agent (dynamic constraints - leading to a parameterization of the set of constraints with respect to the current state).

From the implementation point of view, the pure non-convex optimization solvers can be employed for the control problems mentioned here, but, in our opinion, they are prone to numerical errors and difficult to handle. As such, we find appropriate to use *Mixed-Integer Programming* (MIP) [Jünger et al., 2009] techniques, which have the advantage of *explicitly* including non-convex constraints and discrete decisions in the optimization problem [Osiadacz, 1990]. More than that, they have proven their usefulness in various

applications. Among them, we cite [Richards et al., 2002] for task allocation and trajectory planning for multiple agents, [Prodan et al., 2012c], [Prodan et al., 2011b] for task assignment with coordinated control of multiple agents, subject to dynamics and collision avoidance constraints, and [Stoican et al., 2012] for fault detection and isolation. Also, in [Bemporad and Morari, 1999], [Bemporad et al., 2000], the authors used a combination of MIP and MPC to stabilize general hybrid systems around equilibrium points.

The MIP is not the ultimate solution for the non-convex optimization problems: a sensitive aspect of MIP techniques is the computational complexity which can increase exponentially with the number of binary variables used in the problem formulation. There are some contributions in the literature where the original decision problems are reformulated in a simplified MIP form [Earl and D'Andrea, 2005], [Vitus et al., 2008], but the complexity still remains significant. Other results try to reduce the number of binary variables, e.g., a logarithmic number of binary variables is recapitulated in [Vielma and Nemhauser, 2011].

In the last part of the present chapter, of particular focus will be the MIP description of non-convex regions. We believe that some geometric insights are helpful to reduce the conservatism of existing results. Such an approach was adopted in [Prodan et al., 2012e]. Furthermore, we point out the use of hyperplane arrangements which provide a formal way of describing a *non-connected and non-convex* feasible region. Some of the noteworthy aspects of the approach can be resumed as follows:

- convex representation in the extended space of state plus binary variables using the associated hyperplane arrangement;
- reduced complexity of the problem using boolean algebra techniques (the problem complexity will require only a polynomial number of subproblems (Linear Programming (LP) or Quadratic Programming (QP) problems) that have to be solved with obvious benefits for the computational effort);
- notable property of optimal association between regions and their binary representation leading to the minimization of the number of constraints.

2.1 Optimization-based control

Optimization-based control in general terms refers to the control design using an optimization criterion and the respective resolution techniques in order to obtain the parameters of the control law, the optimality being generally equivalent to a certain desired property as for example, stability, reactivity or robustness. This rather broad definition can cover the classical optimal control, the LMI-based techniques, model predictive control or interpolation-based techniques. In the present work, this terminology refers

to the use of *on-line*, optimal cost function as part of the feedback stabilization of a (typically nonlinear) system [Murray, 2009]. A widely used optimization-based control technique in this class is Model Predictive Control (MPC)[1] also called, *receding horizon control* [2].

The idea behind MPC is to exploit in a receding manner the simplicity of the open-loop optimization-based control [Olaru et al., 2009]. The control action $u(k)$ for a given state $x(k)$ is obtained from the control sequence $\mathbf{u} \triangleq \left\{u(k|k), u(k+1|k), \ldots, u(k+N_p-1|k)\right\}$ as the result of the optimization problem:

$$\arg\min_{\mathbf{u}} \quad V_f(x(k+N_p|k)) + \sum_{s=0}^{N_p-1} V_n(x(k+s|k), u(k+s|k)), \tag{2.1}$$

$$\text{subject to:} \quad \begin{cases} x(k+s+1|k) = f(x(k+s|k), u(k+s|k)), & s = 0, \ldots, N_p-1, \\ h(x(k+s|k), u(k+s|k)) \leq 0, & s = 0, \ldots, N_p-1, \\ h_f(x(k+N_p|k)) \leq 0, \end{cases} \tag{2.2}$$

over a finite horizon N_p. The cost function is comprised of two basic ingredients; namely a terminal cost function $V_f(\cdot) : \mathbb{R}^n \to \mathbb{R}$ and a cost per stage function $V_n(\cdot) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$. Typically, in MPC, the objective (or cost) function (2.1) penalizes deviations of the states and inputs from their reference values, while the constraints are treated *explicitly* [Mayne et al., 2000]. By solving (2.1), each optimization generates an open-loop optimal control trajectory taking into account the system dynamics described by $f(\cdot,\cdot)$, generally with the additional property that f(0,0)=0, the constraints on the states and control inputs $h(\cdot)$ and terminal constraint $h_f(\cdot)$. Then, applying only the first part of the trajectory to the system, based on measurement and recomputing, results in closed-loop control (see, Figure 2.1 which illustrates very well the receding horizon strategy).

The restrictions (2.2) of the optimization problem (2.1) can be written in a more explicit form, stated in terms of hard constraints on the internal state variables and input control action (whenever these are separable):

$$\begin{cases} x(k+s+1|k) = f(x(k+s|k), u(k+s|k)), & s = 0, \ldots, N_p-1, \\ x(k+s|k) \in \mathcal{X}, & s = 0, \ldots, N_p-1, \\ u(k+s|k) \in \mathcal{U}, & s = 0, \ldots, N_p-1, \end{cases} \tag{2.3}$$

where, usually, $\mathcal{X}$ is a convex, closed subset of $\mathbb{R}^n$ and $\mathcal{U}$ is a convex, compact subset of $\mathbb{R}^m$, each set containing the equilibrium point in their strict interior (generally $f(0,0) = 0$ and $0 \in \mathcal{X}$, $0 \in \mathcal{U}$). A terminal constraint could also be imposed for stability reasons

[1] The terminology "Model Predictive" comes from the use of the model to predict the system behavior over the planning horizon at each update.

[2] The terminology "receding horizon" comes from the fact that the planning horizon, which is typically fixed, moves ahead in time with each update.

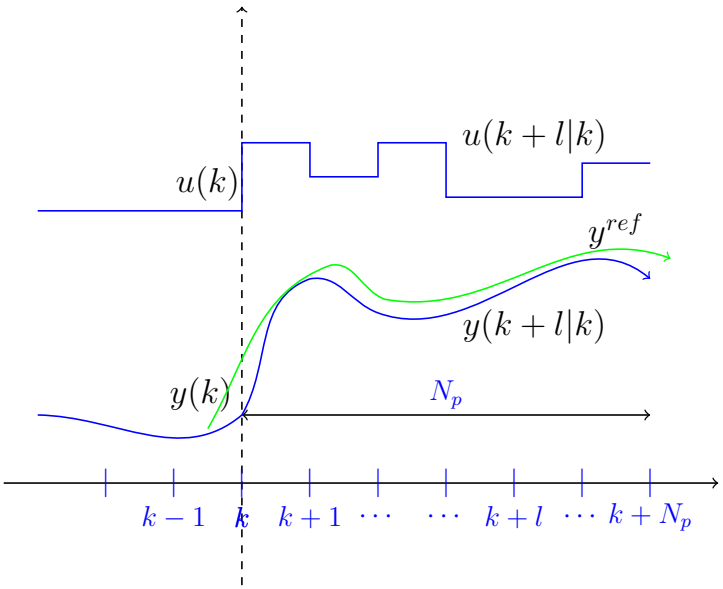


FIGURE 2.1: Receding horizon control philosophy.

[Rawlings and Mayne, 2009]:

$$x(k+N_p|k) \in \mathcal{X}_f \subset \mathcal{X}. \tag{2.4}$$

MPC represents one of the few methods in control that can handle generic state and control constraints. More precisely, it has the ability to include generic models (i.e., nonlinear and linear) and constraints in the optimization-based control problem (2.1)–(2.2). In addition, to this main advantage, it is worth mentioning its' capacity to redefine the cost function and the constraints to account for the changes in the system and/or the environment. In our opinion, the major inconvenient of receding horizon strategy is represented by the computational demand, i.e., it is prohibitive the requirement that an optimization algorithm must run and terminate at every update of the controller block (generally synchronous with the sampling clock).

The first solution to the stability problem in MPC was provided for the unconstrained case, where stability is achieved by having a sufficiently long prediction horizon [Garcia et al., 1989]. Actually, an interesting link has been provided in the MPC understanding by the fact that the unconstrained predictive control problem can be solved by using Linear Quadratic (LQ) or Linear Quadratic Gaussian (LQG) control techniques [Bitmead et al., 1991]. In this direction, the stability theory for the constrained case is based on the existence of an invariant terminal set [Gilbert and Tan, 1991] and the use of a cost function as a Lyapunov function [Vassilaki et al., 1988], [Rossiter, 2003], for the closed loop system. The terminal invariant set is also called the target set for the system

trajectory over the prediction horizon [Chmielewski and Manousiouthakis, 1996]. A survey on the predictive control stability theory can be found in [Mayne et al., 2000], [Goodwin et al., 2005], [Rawlings and Mayne, 2009]. Here, the control policy consists in two distinctive parts:

- to ensure constraints satisfaction and evolution towards an invariant set[3];
- to find some local control law satisfying the constraints and ensuring the positive invariance of the terminal set with respect to the closed-loop dynamic system.

The interdependence between feasibility, invariant sets and stability of constrained predictive control is summarized by the following theorem providing sufficient stability conditions [Mayne et al., 2000]:

Theorem 2.1. *For a given optimal control law* $\boldsymbol{u}(x(k))$ *obtained by solving the optimization problem:*

$$\arg\min_{\boldsymbol{u}} \quad V_f(x(k+N_p|k)) + \sum_{s=0}^{N_p-1} V_n(x(k+s|k), u(k+s|k)), \tag{2.5}$$

$$\textit{subject to:} \quad \begin{cases} x(k+s+1|k) = f(x(k+s|k), u(k+s|k)), & s = 0, \dots, N_p-1, \\ x(k+s|k) \in \mathcal{X}, & s = 0, \dots, N_p-1, \\ u(k+s|k) \in \mathcal{U}, & s = 0, \dots, N_p-1, \\ x(k+N_p|k) \in \mathcal{X}_f \subset \mathcal{X}, \end{cases} \tag{2.6}$$

the feasibility at time instant 0 *implies the feasibility at all future time instances and the stability of the closed-loop system* $x(k+1) = f(x(k|k), \boldsymbol{u}(x(k)))$ *is guaranteed if:*

1. $f(\cdot,\cdot)$, $V_n(\cdot,\cdot)$ *and* $V_f(\cdot)$ *are continuous;*
2. $f(0,0) = 0$, $V_n(0,0) = 0$ *and* $V_f(0) = 0$*;*
3. $\mathcal{U}$ *is compact and contains the origin;*
4. $\mathcal{X}$ *is closed and contains the origin in its strict interior;*
5. $\mathcal{X}_f$ *is closed and contains the origin in its strict interior;*

□

In the present work the constraints represent some dominating factor to be appropriately taken into account. Consequently, based on the considerations above, we find appropriate the use of MPC as a design method for the control of Multi-Agent Dynamical Systems.

[3]Invariance notions will be presented in the forthcoming sections.

In these particular settings, we identify four main ingredients in the Model Predictive Control techniques:

- the model of the system used for prediction purposes;
- the reference trajectory[4];
- the constraint sets representation;
- the optimization solver.

Each one of the necessary ingredients will be detailed in the forthcoming sections in order to gain an insight of the elements needed at the design stage.

2.2 A generic prediction model

In the present section, we present different models used throughout the manuscript to predict the behavior of individual agents. The model of the systems used for the prediction purposes, can be either continuous-time or discrete-time, either homogeneous or heterogeneous, either linear or non-linear and/or affected by disturbances. We introduce here the discrete-time case for uniformity of notation, but the continuous-time case can be also used under certain conditions (unless explicitly stated, the discrete-time models will be exploited in the optimization problems). Indeed, the receding horizon control builds on discrete-time optimal control problem solved at each sampling instant, therefore, we need to be able to handle the discretized and linearized model of the systems. In many situations throughout the manuscript we will restrict ourselves to linear time-invariant models and nonlinear unicycle models.

First, consider the following discrete-time autonomous system:

$$x(k+1) = f(x(k)), \quad x(k) \in \mathcal{X}, \tag{2.7}$$

where $x(k) \in \mathbb{R}^n$ is the current state and the mapping $f(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ is assumed to be continuous on $\mathbb{R}^n$ satisfying the condition $f(0) = 0$. The state constraint set $\mathcal{X}$ is a compact set containing the origin in its interior.

Second, adding additive disturbances to the previous model (2.7), we have:

$$x(k+1) = f(x(k), w(k)), \quad (x(k), w(k)) \in \mathcal{X} \times \mathcal{W}, \tag{2.8}$$

[4]Note that the second item is not mandatory, it becomes so only in tracking problems as for example, in the context of multi-agent systems with trajectory tracking objectives. Alternatively, this trajectory is replaced by a set-point (an equilibrium point, the origin in most of the LTI regulation studies).

where, the disturbance $w(k)$ is bounded, i.e. $w(k) \in \mathcal{W}$ and $\mathcal{W} \subset \mathbb{R}^p$ is a convex and compact set containing the origin. The function $f(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ is assumed to be continuous and satisfying the condition $f(0,0) = 0$.

Consider also the following discrete-time invariant system:

$$x(k+1) = f(x(k), u(k)), \quad (x(k), u(k)) \in \mathcal{X} \times \mathcal{U}, \tag{2.9}$$

where, in addition to the first system, $u(k) \in \mathbb{R}^m$ is the current control input and the mapping $f(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is assumed to be continuous with $f(0,0) = 0$. The control constraint set $\mathcal{U}$ is a compact sets containing the origin in its interior.

Lastly, the discrete-time, time-invariant dynamics affected by additive disturbances is written in the following form:

$$x(k+1) = f(x(k), u(k), w(k)), \quad (x(k), u(k), w(k)) \in \mathcal{X} \times \mathcal{U} \times \mathcal{W}. \tag{2.10}$$

The function $f(\cdot,\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}^n$ is assumed to be continuous, $f(0,0,0) = 0$ and the sets $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{W}$ are compact sets and each of them contains the origin in their respective interior.

In the following section we provide a short description of the technique we choose for the generation of a reference trajectory, the second necessary ingredient in the MPC formulation.

2.3 Generation of a reference trajectory

One of the important issues arising in multi-agent motion planning is "how to get there?" or "what is the appropriate pattern to align to?". The answer is given by the resolution of a trajectory planning problem that is. in principle, an open-loop control problem. The objective here is to generate a real time trajectory to guide a group of agents to their destination while avoiding their neighboring agents and other possible obstacles. We make the difference here between the reference trajectory and reference path. The reference path only provides a desired geometric route for the agents, without a timing control law assigned to it. The reference trajectory has a time representation, in the sense that it provides simultaneously both input and state signals. In the present work we choose appropriate differential flatness tools to derive input and state reference trajectories.

The flatness was studied as a generalization of the structural properties of the linear systems, which exhibit a state representation obtained via derivatives of the input and output signals. The class of systems that exhibit the property of differential flatness were first studied by [Fliess et al., 1995]. Other contributions to the topic were presented in [Rouchon et al., 2003], [Sira-Ramírez and Agrawal, 2004], [Lévine, 2009], [De Doná

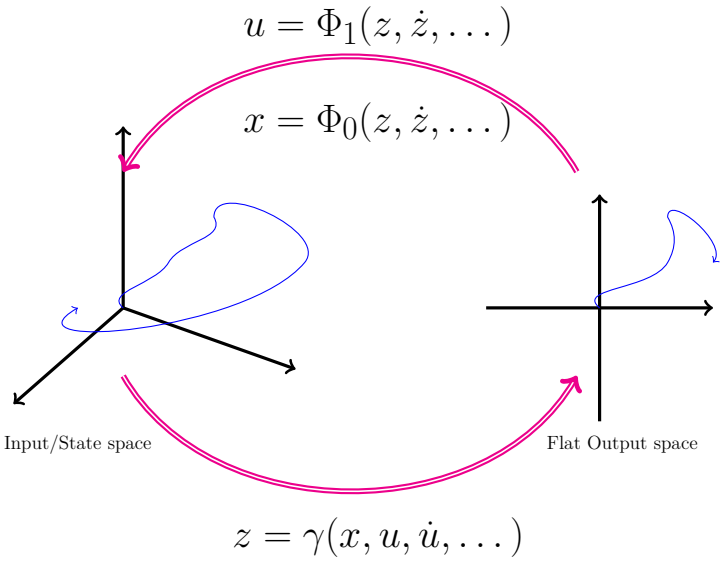


FIGURE 2.2: Differentially flat systems.

et al., 2009]. The interested reader is referred also to [Murray, 1997] for a description of the role of flatness in control of mechanical systems in close relation with the way these properties will be employed in the present work. In the sequel, we provide a brief overview of differential flatness based primarily on the technical report of [Murray, 2009] and the monograph of [Lévine, 2009].

Consider the general system[5]:

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0, \; x(T) = x_f, \tag{2.11}$$

where $x(t) \in \mathbb{R}^n$ is the state vector and $u(t) \in \mathbb{R}^m$ is the input vector.

Definition 2.1. *The system (2.11) is called differentially flat if there exist variables $z(t) \in \mathbb{R}^m$ such that the states and inputs can be algebraically expressed in terms of $z(t)$ and a finite number of its higher-order derivatives:*

$$\begin{aligned} x(t) &= \Phi_0(z(t), \dot{z}(t), \cdots, z^{(q)}(t)), \\ u(t) &= \Phi_1(z(t), \dot{z}(t), \cdots, z^{(q)}(t)), \end{aligned} \tag{2.12}$$

[5]We note that models (2.9) and (2.11) describe similar dynamics, one in discrete time, and the other, in continuous time. The correspondence between the continuous and the discrete signals can be made by a large class of discretization techniques [Deuflhard et al., 1987].

where $z(t) = \gamma(x(t), u(t), \dot{u}(t), \cdots, u^{(q)}(t))$ *is called the flat output*[6] *and* q *is the maximum order of* $z(t)$ *arising in the problem.* □

Remark 2.1. One of the observation made in the differential flatness literature is that, for any linear and nonlinear flat system, the number of flat outputs equals the number of inputs [Lévine, 2009] (see also Figure 2.2 which simply illustrates the concept of flatness). □

Remark 2.2. Note also that (as discussed in [Sira-Ramírez and Agrawal, 2004]), for linear systems, the flat differentiability (existence and constructive forms) is implied by the controllability property. □

Differentially flat systems are well suited to problems requiring trajectory planning as for example MPC which builds on a prediction capability over a finite horizon. The most important aspect of flatness is that it reduces the problem of trajectory generation for the original system modeled by an ordinary differential equation, to finding a trajectory of the flat outputs via the resolution of an algebraic system of equations. The trajectories are planned for $z(t)$ and then, transformed via $\Psi_0(\cdot)$ and $\Phi_1(\cdot)$ in (2.12) to obtain the trajectories of states and inputs. The most important aspect from the subsequent feedback control is that these trajectories are consistent with the system dynamic equation. Practically, any trajectory for $z(t)$ satisfying the boundary conditions:

$$\begin{aligned} x(0) &= \Phi_0(z(0), \dot{z}(0), \cdots, z^{(q)}(0)) &&= x_0, \\ x(T) &= \Phi_0(z(T), \dot{z}(T), \cdots, z^{(q)}(T)) &&= x_f, \\ u(0) &= \Phi_1(z(0), \dot{z}(0), \cdots, z^{(q)}(0)) &&= u_0, \\ u(T) &= \Phi_1(z(T), \dot{z}(T), \cdots, z^{(q)}(T)) &&= u_f, \end{aligned} \tag{2.13}$$

will be a feasible trajectory for the system that passes through the given initial and final conditions.

One objection that can be raised is that equation (2.13) still contains the derivatives of $z(t)$ and as such is not purely algebraic expression. However, the flat output in (2.13) can be parameterized using a set of smooth basis functions $\Lambda^i(t)$ in order to allow the derivatives manipulation:

$$z(t) = \sum_{i=1}^{N_\alpha} \alpha_i \Lambda^i(t), \quad \alpha_i \in \mathbb{R}. \tag{2.14}$$

The set of coefficients α_i in (2.14), with $i = 1, \cdots, N_\alpha$ represents practically the unknown variables in the system described by the equations (2.13) which lead in this specific form (2.13)–(2.14) to a system of linear equations. In this framework the derivatives of the

[6]The definition proposed here do not provide an insight on the existence conditions and the related qualitative problems that can appear in the definition of the nonlinear functions. We assume that the characteristics necessary for the existence of a flat trajectory (practically controllability) are respected for the manipulated dynamical system.

flat output can be computed in terms of the derivatives of the basis functions:

$$\begin{aligned} \dot{z}(t) &= \sum_{i=1}^{N_\alpha} \alpha_i \dot{\Lambda}(t), \\ &\vdots \\ z^{(q)}(t) &= \sum_{i=1}^{N_\alpha} \alpha_i \Lambda^{(q)}(t). \end{aligned} \tag{2.15}$$

Subsequently, the conditions on the flat outputs and their derivatives are written as follows:

$$\begin{bmatrix} \Lambda_1(0) & \Lambda_2(0) & \cdots & \Lambda_{N_\alpha}(0) \\ \dot{\Lambda}_1(0) & \dot{\Lambda}_2(0) & \cdots & \dot{\Lambda}_{N_\alpha}(0) \\ \vdots & \vdots & & \vdots \\ \Lambda_1^{(q)}(0) & \Lambda_2^{(q)}(0) & \cdots & \Lambda_{N_\alpha}^{(q)}(0) \\ \Lambda_1(T) & \Lambda_2(T) & \cdots & \Lambda_{N_\alpha}(T) \\ \dot{\Lambda}_1(T) & \dot{\Lambda}_2(T) & \cdots & \dot{\Lambda}_{N_\alpha}(T) \\ \vdots & \vdots & & \vdots \\ \Lambda_1^{(q)}(T) & \Lambda_2^{(q)}(T) & \cdots & \Lambda_{N_\alpha}^{(q)}(T) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{N_\alpha} \end{bmatrix} = \begin{bmatrix} z(0) \\ \dot{z}(0) \\ \vdots \\ z^{(q)}(0) \\ z(T) \\ \dot{z}(T) \\ \vdots \\ z^{(q)}(T) \end{bmatrix}. \tag{2.16}$$

The formulation (2.16) takes the form of a system of *linear* equation, which can be rewritten in the form $M\alpha = \bar{z}$. Finally, assuming that M has a sufficient number of columns and it is a full column rank (classical well-possedness conditions), the trajectory generation problem is solved by finding the coefficient α (possibly non-unique).

Remark 2.3. Note that, the equations (2.13) represent typical conditions which appear in flat trajectory generation and can be solved as a classical system of equations as a function of parameters in the generation of the signal $z(t)$. This basic boundary problem can be modified and enhanced in several ways, if necessary. Firstly, additional constraints can be applied upon the derivatives of the state and the input, thus reducing the feasible set of solutions. Finally, we can consider conditions not only at the initial and the final times but also at any intermediate time instant, by increasing the number of equations (and adequately adjusting the number of parameters in order to ensure the well-posedness) [Wilkinson, 1965]. □

Note that in (2.16) the conditions on the initial and final time instants can be extended for simplicity to the introduction of a set of way-points through which the flat trajectory must pass (as also mention in Remark 2.3). This in equivalent to the introduction of additional rows in (2.16). Since a continuous trajectory (and smooth, usually) is needed, there are two ways of accomplishing it in the current framework. One is to apply (2.13)–(2.16) as they are, and then force the resulting "segments" to link (by matching conditions on the intermediary points) or, otherwise, as mentioned in Remark 2.3, to solve everything as a single problem where we add conditions on intermediate time

instants. As it can be seen, both approaches have shortcomings and strengths which will be discussed on a practical example in Section 5.1.

For the global trajectory planning it is important to point out that, there are different basis functions used for parametrization of the flat output ($\Lambda^i(t)$ in (2.14)), each with their advantages and disadvantages.

- A class of functions commonly used is represented by *polynomials* in the time variable– t, which are capable of representing curves in a flexible way. However, they have a poor numerical performance, their dimension depends on the number of conditions imposed on the inputs, states and their derivatives. Moreover, small variations in their coefficients may affect the curve's shape in ways impossible to foretell(see for more details on the numerical performance and comparison with other methods [Farouki and Rajan, 1987], [Daniel and Daubisse, 1989], [Suryawan, 2012]). These are known difficulties in the numerical methods associated to linear algebra, the interesting reader may refer to conditioning problems in Vandermonde systems of equations [Demmel and Koev, 2005].

- A different class of polynomials for representing a curve is given by *Bésier basis functions*. The equivalence in terms of degrees of freedom is enhanced from the numerical condition point of view. This equivalence ensures that one representation can be translated into the other without loss of information [Farin, 2001], [Yamaguchi, 1988]. The limitation of Bésier curves is related again to the need of higher degrees to satisfy a large number of inequality and/or equality constraints (for more details, see for instance [Suryawan, 2012]).

- It is more convenient to represent the flat output $z(t)$ using *B-splines* [Schumaker, 2007], because of their ease of enforcing continuity across way-points and ease of computing their derivatives. Moreover, their degree depends only up to which derivative is needed to ensure continuity, this being in contrast with the polynomial basis functions previously mentioned. The studies [De Doná et al., 2009], [Suryawan et al., 2010], [Suryawan, 2012] developed a parameterization based on B-spline curves for the flat output matching thus the properties of differentially flat linear systems. Based on this fact, they also considered the problem of generating constrained flat trajectories for receding horizon control.

2.4 Set-theoretic elements

For a robust approach we invoke the use of set-theoretic methods in the feedback control design. Whenever the noises and disturbances affecting the systems under consideration are bounded, they can be represented set-wise and consequently bounded invariant sets which characterize their dynamics can be practically constructed. For example, we

can construct safety regions around agents in order to guarantee collision avoidance for any values of the bounded noises or reachable sets for reaching a target position for a formation.

The foundation of set-theoretic methods in control are inherited from the well-developed mathematical set theory, particularly from the Brunn-Minkowski algebra [Schneider, 1993], [Aubin et al., 2011]. Their role has been already discussed in the constraint control synthesis (see for example the tube MPC) [Mayne et al., 2005], [Rakovic et al., 2011], [Artstein and Raković, 2008], the characterization of the minimal invariant sets [Raković et al., 2005], [Kouramas et al., 2005] and also fault tolerant control [Seron et al., 2008], [Stoican et al., 2011a], [Stoican, 2011], [Stoican et al., 2012]. The elements we refer to in the present manuscript are the positive and control invariance in the presence of disturbances (see, for instance, the comprehensive paper survey of [Blanchini, 1999] and the monography [Blanchini and Miani, 2008]). In particular, we are interested in characterizing the positive invariance of a subset of the state-space of a dynamical system [Bitsoris, 1988], [Bitsoris and Truffet, 2006]. Also, we are interested in their construction using ultimate bounds notions [Kofman et al., 2007b] and robust positive invariant set representation [Raković, 2007]. Furthermore, the reachable set computation [Bertsekas, 1995] will be useful in characterizing a visibility region around agents in the context of a distributed MPC schema.

We can choose between various classes of sets in control, their strengths and weaknesses being primarily related to the numerical representation and computational complexity. A standard class is represented by the ellipsoidal sets. Due to their simple numerical representation, they are extensively used in a broad variety of applications [Kurzhanskiĭ and Vályi, 1997]. Their main inconvenient is represented by their conservatism when it comes to algebraical operations (intersection, convex hull and other). Among other classes of sets, we mention the Linear/Bilinear Matrix Inequalities (LMI/BMI), that appears to be more attractive in the structure flexibility to the ellipsoidal sets [Nesterov and Nemirovsky, 1994].

For numerical reasons, the family of sets preferred in the present work is that of polyhedral sets. They provide a good balance between flexibility of representation and numerical implementation of the algorithms [Blanchini, 1999], [Motzkin et al., 1959]. More precisely, they can be classified somewhere in the "middle", that is, they are flexible in the sense that they can approximate arbitrary well any convex shape and numerically manageable in the sense that they are relatively[7] easy to use (mainly due to their dual formulation, both in half-space and vertex form [Fukuda, 2004]).

In the forthcoming chapters, we will make use of the polyhedral sets for describing safety regions around agents, obstacles and corresponding feasible regions.

[7]Note that the polyhedral sets are relatively easy to use in low dimensional systems via their dual representation. For high dimensions, the double-representation algorithms scale badly, suffering from the course of dimensionality.

Finally, the link between set-theoretic analysis and MPC becomes straightforward by using the above elements, i.e., (robust) positively invariant sets, reachable sets, terminal sets and feasible sets.

In order to provide some formal basis for the set manipulations, we present first the polyhedral sets description. Starting from their representation, we will further define polyhedral function and sum function which will represent the basis for the construction of "exclusion" functions (or penalty functions) in the forthcoming chapter. Finally, a particular emphasize will be put on set-theoretic concepts in control and invariance notions.

2.4.1 Polyhedral sets description

Let us define a bounded convex set in its polyhedral approximation, a polytope $S \subset \mathbb{R}^n$ through the implicit half-space description:

$$S = \left\{ x \in \mathbb{R}^n : h_i x \leq k_i, \quad i = 1, \ldots, N \right\}, \tag{2.17}$$

with $(h_i, k_i) \in \mathbb{R}^{1 \times n} \times \mathbb{R}$ and N the number of half-spaces. We focus on the case where $k_i > 0$, meaning that the origin is contained in the strict interior of the polytopic region, i.e. $0 \in Int(S)$.

By definition, every supporting hyperplane for the set S in (2.17):

$$\mathcal{H}_i = \{x : \ h_i x = k_i\} \tag{2.18}$$

will lead to a partition of the space into two disjoint[8] regions:

$$\mathcal{R}^+(\mathcal{H}_i) = \{x : \quad h_i x \leq \quad k_i\}, \tag{2.19}$$

$$\mathcal{R}^-(\mathcal{H}_i) = \{x : \ -h_i x \leq -k_i\}, \tag{2.20}$$

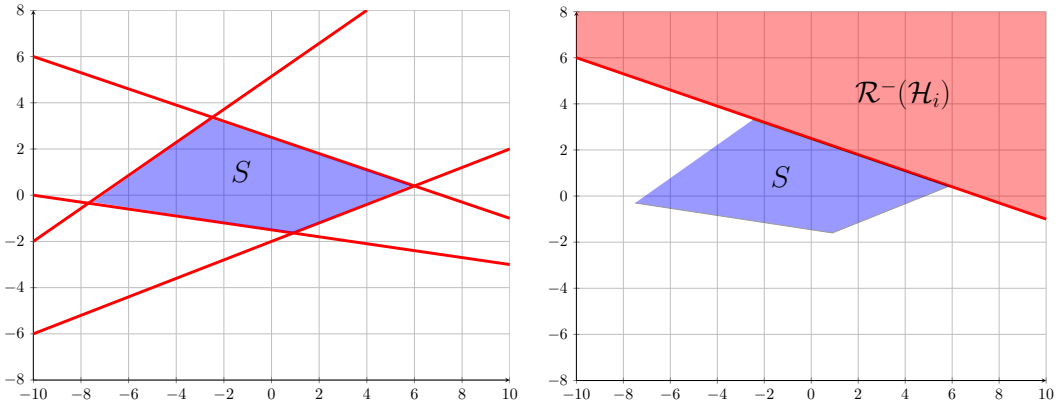
with $i = 1, \ldots, N$. Here $\mathcal{R}_i^+$ and $\mathcal{R}_i^-$ denote, in a simplified formulation, the complementary regions associated to the i^{th} inequality of (2.17). In order to have the ideas clear, the above relations can be represented as in Figure 2.3.

Let us also recall a general result relative to convex sets which will be used in the optimization problems of the forthcoming chapters:

Proposition 2.1. *For any two convex sets $S_1, S_2 \in \mathbb{R}^n$ the next relations are equivalent:*

1. $S_1 \cap S_2 = \emptyset$

[8]The relative interiors of these regions do not intersect but their closures have as common boundary the affine subspace $\mathcal{H}_i$.



(A) Polytope S. (B) Polytope S and $\mathcal{R}^-(\mathcal{H}_i)$.

FIGURE 2.3: Bounded convex set S.

2. $\{0\} \notin S_1 \oplus \{-S_2\}$.

□

Proof: It suffices to note that if the origin is inside the set $S_1 \oplus \{-S_2\}$ then, necessarily, there exists $x_1 \in S_1$ and $x_2 \in S_2$ such that $x_1 - x_2 = 0$. □

In the sequel, using the representation of the bounded polyhedral set in (2.17), we will define the polyhedral function and sum function such that they preserve the shape of the polyhedral set.

2.4.2 Polyhedral function

Consider the class of (symmetrical) piecewise linear functionals defined using the specific shape of a polyhedral set. The following definitions will be instrumental in the present section.

Definition 2.2 (Minkowski function – [Blanchini, 1995])**.** *Any bounded convex set S induces a Minkowski function $\mu(x) : \mathbb{R}^n \to \mathbb{R}$ defined as:*

$$\mu(x) = inf \left\{ \alpha \in \mathbb{R},\ \alpha \geq 0 :\ x \in \alpha S \right\} \tag{2.21}$$

□

Definition 2.3 (Polyhedral function – [Blanchini, 1995]). *A polyhedral function is the Minkowski function of the polyhedral bounded convex set S defined in (2.17). This function $\mu(x) : \mathbb{R}^n \to \mathbb{R}$ has the following expression:*

$$\mu(x) = \|Fx\|_\infty, \tag{2.22}$$

where $F \in \mathbb{R}^{N\times n}$ is a full column matrix with $F_i = \frac{h_i}{k_i}$, $i = 1, \ldots, N$. □

In fact, any polytope can be defined in terms of the Minkowski function (2.21). Indeed, there always exists a full column matrix $F \in \mathbb{R}^{N\times n}$ such that the corresponding polytope S in (2.17) is equivalently defined as:

$$S = \left\{ x \in \mathbb{R}^n : \mu(x) \leq 1 \right\}, \tag{2.23}$$

with $\mu(x)$ defined by (2.22). From the avoidance point of view, the Minkowski function (2.21) denotes the inclusion of a value x to the given polytope (2.23) if $\mu(x) \in [0, 1]$. Conversely, if $\mu(x) > 1$ then x is outside the polytope (2.23).

Remark 2.4. Note that if $k_i \leq 0$ in (2.17), the origin is not contained in the strict interior of the polytopic region, i.e. $0 \notin Int(S)$, then the polyhedral function can be brought to the form (2.22) by imposing:

$$F_i = \frac{h_i(x - x_s)}{k_i - h_i x_s}, \quad i = 1, \ldots, N, \tag{2.24}$$

with $x_s \in \mathbb{R}^n$ the analytic center of the polytope (2.17). □

Notice that the polyhedral function (2.22) is piecewise affine and continuous. This means that each of the inequalities composing its definition can provide the maximum, an explicit description of these regions being of the form:

$$X_i = \left\{ x \in \mathbb{R}^n : \tfrac{h_i}{k_i} x > \tfrac{h_j}{k_j} x, \forall\, i \neq j,\ i, j = 1, \ldots, N \right\}. \tag{2.25}$$

The entire space can thus be partitioned in a union of disjoint regions X_i which are representing in fact cones with a common point in the origin (respectively in x_s for the general case evoked in Remark 2.4).

Practically, the polyhedral function (2.22) can be represented in the form:

$$\mu(x) = F_i x, \quad \forall\, x \in X_i, \quad i = 1, \cdots, N. \tag{2.26}$$

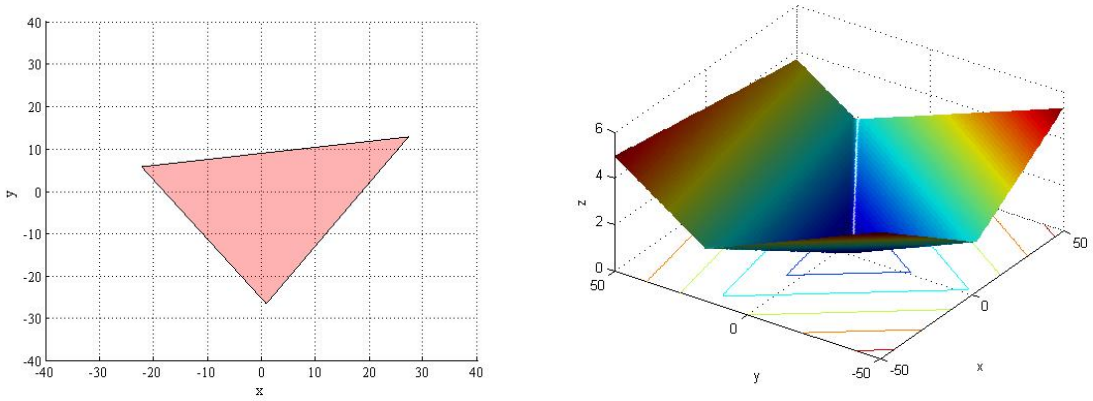
The piecewise affine gradient of (2.26) is defined as:

$$\bigtriangledown \mu(x) = F_i, \quad \forall\, x \in X_i, \quad i = 1, \ldots, N. \tag{2.27}$$

and will be referred in the forthcoming chapter upon this noation.

Remark 2.5. Strictly speaking, the generalized gradient (2.27) is multivalued (the Minkowski function induced by a polytope is not differentiable in the classical sense, rather it is differentiable almost everywhere). However, an univocal candidate can be selected for the computations and such an approach is used in the rest of the manuscript. We mention that, alternatively, the explicit use of multivalued expression of the gradient would not bring computational difficulties as long as the range of variation is bounded and can be represented by the extreme values in practice. □

An illustrative representation of the above construction can be depicted in Figure 2.4, where we represent first a bounded convex set and then, its polyhedral function defined accordingly to (2.26).



(A) Bounded polyhedral set.

(B) Polyhedral function of the bounded polyhedral set.

FIGURE 2.4: Polyhedral function representation.

2.4.3 Sum function

Consider again the polytope defined in (2.17), and a piecewise linear function (introduced in [Camacho and Bordons, 2004]):

$$\psi(x) = \sum_{i=1}^{N} (h_i x - k_i + |h_i x - k_i|). \tag{2.28}$$

The function (2.28) is zero inside the convex region (2.17) and increases linearly in the exterior, as the distance to the frontier is increased.

The definition (2.28) describes in fact a continuous piecewise affine function over a partition of the state-space. Over each of the polyhedral cells composing this partition,

the absolute values of $|h_i x - k_i|$ are constant resulting in a fixed affine form for $\psi(x)$. In order to explicitly describe the regions composing the partition, additional theoretical notions related to the arrangements of the hyperplanes will be introduced in the sequel. The interested reader may consult on advanced or classical results on this topic in the well known books [Orlik and Terao, 1992] and [Ziegler, 1995], as well as some recent related articles [Orlik, 2009], [Pardalos, 2009] and the references therein.

Definition 2.4 (Hyperplane arrangements – [Ziegler, 1995])**.** *A collection of hyperplanes $\mathbb{H} = \{\mathcal{H}_i\}$, with $a = i, \ldots, N$, partitions the space in an union of disjoint cells defined as follows:*

$$\mathcal{A}(\mathbb{H}) = \bigcup_{l=1,\ldots,\gamma(N)} \underbrace{\left(\bigcap_{i=1,\cdots,N} \mathcal{R}_i^{\sigma_l(i)}(\mathcal{H}_i) \right)}_{A_l}, \tag{2.29}$$

where $\sigma_l \in \{-,+\}^N$ denotes all feasible combinations of regions (2.19) and (2.20) obtained for the hyperplanes in $\mathbb{H}$ and $\gamma(N)$ denotes the number of feasible cells. □

Note that the number of regions in the hyperplane arrangement is usually much greater than the number of regions (2.25) associated to the polyhedral function (2.26). Therefore, the piecewise affine function (2.28) can be alternatively described as:

$$\psi(x) = 2 \sum_{\substack{i=1 \\ \sigma_l = \{+\}}}^{N} (h_i x - k_i), \ \forall\ x \in \mathcal{A}_l, \ l = 1, \ldots, \gamma(N). \tag{2.30}$$

The piecewise affine gradient of (2.30) is defined as:

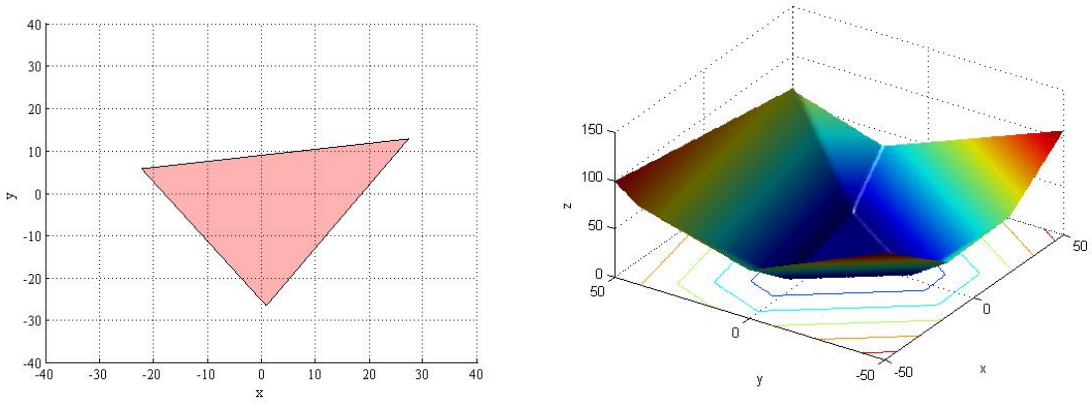
$$\bigtriangledown \psi(x) = 2 \sum_{\substack{i=1 \\ \sigma_l = \{+\}}}^{N} h_i^T, \ \forall\ x \in \mathcal{A}_l, \quad l = 1, \ldots, \gamma(N). \tag{2.31}$$

As in the previous case, we illustrate in Figure 2.5 the same bounded convex set and the sum function defined accordingly to (2.30).

2.4.4 Characterization and construction of invariant sets

For future use, we define several fundamental notions associated to sets in control. We will use the models description given in Section 2.2 or a simplified form of them.

Definition 2.5. *[Blanchini, 1999] A set $S \in \mathbb{R}^n$ is called positively invariant for a system of the form (2.7) if for all $x(0) \in S$ the solution, $x(k) \in S$, for $k > 0$. If $x(0) \in S$ implies $x(k) \in S$ for all $k \in \mathcal{Z}$, then we say that S is invariant.* □



(A) Bounded polyhedral set.

(B) Sum function of the bounded polyhedral set.

FIGURE 2.5: Sum function representation.

In other words, a set S is (positively) invariant if it is guaranteed that, if the current state lies within S, all future states also lies within S. For example, any *equilibrium point* is an invariant set as long as, in the absence of external perturbation, the state remains at equilibrium at all future instants. The domain of attraction of an equilibrium point is also an invariant set upon the same argument.

Definition 2.6. *[Blanchini, 1999] A set S is called Robustly Positively Invariant (RPI) for a system of the form (2.8) if for all $x(0) \in S$ and all $w(k) \in \mathcal{W}$ the solution is such that $x(k) \in S$ for $k > 0$.* □

Remark 2.6. Note that in general, when the control law is not a priori fixed, we have controlled (robust) invariance (see also [Blanchini, 1999]), but this is out of the scope of the present work. □

Another fundamental notion from set theory is the one of the *reachable set*, that is the set that all the trajectories can reach in a predefined number of steps, starting from an initial set and under dynamics (2.8). A formal definition along the lines in [Girard et al., 2006] follows.

Definition 2.7. *Consider an initial set S. Then, the k-step reachable set from S under the dynamics (2.8) is denoted as $R_k(f(x(k), w(k)), S)$ and is given by the recursive relation:*

$$R_i = f(R_{i-1}, w), \ \forall i = 1, \ldots, k, \ \text{and} \ \ R_0 = S. \tag{2.32}$$

Definition 2.8. *[Blanchini and Miani, 2007] A set Ω_∞ is called minimal Robustly Positively Invariant (mRPI) for a systems of the form (2.8) if it is a RPI set in $\mathbb{R}^n$ contained in every RPI set for (2.8).* □

Remark 2.7. Note that the mRPI set can be seen as the ∞-step reachable set starting from $\{0\}$.

There are various algorithms able to offer arbitrary close outer approximation of the mRPI set associated to (2.8) (as, for example, the approaches proposed by [Raković et al., 2005], [Olaru et al., 2008]). It is worth mentioning that these algorithms avoid the exponential increase in the complexity of the representation. To overcome this inconvenience, an ultimate bound construction (as proposed in [Kofman et al., 2007a]) can be used, due to its low computational effort.

The following theorem from [Kofman et al., 2007a] is recalled here as an instrumental result for the linear class of systems (2.8). Furthermore, we can express the system dynamics (2.8) in a simplified form:

$$x(k+1) = Ax(k) + w, \quad w \in \mathcal{W}, \tag{2.33}$$

where A is assumed to be diagonalizable and stable.

Theorem 2.2. *[Kofman et al., 2007a] Consider the system (2.33) and let $A = VJV^{-1}$ be the Jordan decomposition of A, with J a diagonal matrix and V an invertible matrix. Consider also a nonnegative vector $\overline{w}$ such that $|w| \leq \overline{w}$ elementwise, $\forall w \in \mathcal{W} \subset \mathbb{R}^n$. Then the set:*

$$\Omega_{UB} = \left\{ x \in \mathbb{R}^n : |V^{-1}x| \leq (I - |J|)^{-1})|V^{-1}|\overline{w} \right\} \tag{2.34}$$

is robustly positive invariant with respect to the dynamics (2.33). □

Note that construction methods can be developed using Theorem 2.2 [Kofman et al., 2007a] and further adapted for nonlinear dynamics [Haimovich et al., 2007], state-dependent perturbations [Kofman et al., 2008], optimizations for implicit/ explicit bounds [Haimovich et al., 2008], generalizations through a perturbation signal [Kofman et al., 2008]. A contribution for ultimate bounds with zonotopic disturbances is presented in [Stoican et al., 2011a], where less conservative constraints are employed.

Finally, the invariance of a set can be equivalent with the feasibility of simple Linear Programming (LP) problem (which makes use of the Farkas Lemma [Ziegler, 1995]), as stated in [Bitsoris, 1988]. The following lemma provides an insight in the relation between linear algebra and set invariance. It holds in the Linear Time Invariant (LTI) case for polytopic sets, but the notions have been extended in order to deal with more general shapes [Loskot et al., 1998], [Bitsoris and Athanasopoulos, 2011].

Lemma 2.1. *[Bitsoris, 1988] The set $R(F, \theta)$ with $F \in \mathbb{R}^{s \times n}$ and $\theta \in \mathbb{R}^s$ is a contractive (positively invariant) set for the autonomous system*

$$x(k+1) = Ax(k), \tag{2.35}$$

if there exists an elementwise positive matrix $H \in \mathbb{R}^{s\times s}$ *and a scalar* $\epsilon \in (0,1]$ *such that:*

$$HF = FA, \tag{2.36}$$
$$H\theta \leq \epsilon\ \theta. \tag{2.37}$$

□

Up to now we have assumed bounded convex shapes. We have introduced notions which are naturally linked with convex shapes (polytopic sets, polyhedral norms, invariance conditions). However, we also need to consider non-convex shapes, as they arise naturally in the control and coordination of multi-agent systems. To this end, we propose next to describe non-connected and non-convex regions by using mixed-integer techniques.

2.5 A contribution for mixed-integer description of non-convex feasible regions

For safety and obstacle avoidance problems (to take just a few examples), the feasible region in the space of solutions is a non-convex set. Usually, this region is considered as the complement of a convex region which describes (contains) an obstacle and/or a safety region, prohibited for the trajectories of the dynamical system.

Considering the definition of the bounded polyhedral set S in (2.17) we define its complement as:

$$\mathcal{C}_X(S) \triangleq cl(X \setminus S), \tag{2.38}$$

with the reduced notation $\mathcal{C}(S)$ whenever X is presumed known or is considered to be the entire space $\mathbb{R}^n$. In an explicit form, the non-convex region $\mathcal{C}(S)$, denoted by (2.38), may then be described as a union of regions that cover all space excepting S:

$$\mathcal{C}(S) = \bigcup_i \mathcal{R}^-(\mathcal{H}_i), \quad i = 1, \ldots, N, \tag{2.39}$$

with $\mathcal{H}_i$ defined as in (2.18). We note that the complement of a bounded polyhedron (2.17) is covered in (2.39) by an union of *overlapping regions* (obtained as elementwise complement of the regions (2.20) associated to the i^{th} inequality of (2.17)).

Even if mathematically self-contained, the definition (2.39) is not attractive for the expression of feasible domains of optimization problems, in order to obtain such a tractable problem formulation, mixed-integer techniques can be used, with the aim of defining a polyhedron in the extended space of *state* and *auxiliary binary variables* of the form:

$$-h_i x \leq -k_i + M\alpha_i, \quad i = 1, \ldots, N, \tag{2.40}$$

$$\sum_{i=1}^{N} \alpha_i \leq N - 1, \tag{2.41}$$

with M a constant chosen appropriately (that is, significantly larger[9] than the rest of the variables in the hyperplane definitions and playing the role of a relaxation constant) and $(\alpha_1, \ldots, \alpha_N) \in \{0,1\}^N$ the auxiliary binary variables (which can activate or not the relaxation).

Remark 2.8. The set of solutions for (2.40)–(2.41) can be projected on the original space $\mathbb{R}^n$, leading to a coverage of the non-convex region, which corresponds to the implicit definition in (2.38). A region $\mathcal{R}_i^-$ can be obtained from (2.40), with an adequate choice of binary variables:

$$\alpha^i \triangleq (1, \ldots, 1, \underbrace{0}_{i}, 1, \ldots, 1). \tag{2.42}$$

Note also that no choice of binary variables can lead to the description of a region $\mathcal{R}_i^+$ as in (2.19). Indeed, if a binary variable is "1", the corresponding inequality degenerates such that it covers any point $x \in \mathbb{R}^n$ (this represents the limit case for $M \to \infty$). The condition (2.41) is thus required to ensure that at least one binary value be "0" and, consequently, at least one inequality be verified. □

As it can be seen in the representation (2.40)–(2.41), the use in an optimization problem is straightforward by the linear expression of the inequalities. However, a binary variable is associated to each inequality in the description of the polytope (2.17). Obviously, for a big number of inequalities, the number of binary variables becomes exceedingly large. Since their number exponentially affects the resolution of any mixed integer algorithm (usually based on *branch-and-cut* techniques and, thus very sensitive to the number of binary terms), the goal to reduce their number is worthwhile. A first step would be to eliminate from the half-space representation of the polytope all the redundant constraints, see Olaru and Dumur [2005]. We suppose that this pre-treatment has been performed, and we are dealing with a non-redundant description of the polyhedral set in (2.17).

2.5.1 Basic ideas for improvements in non-convex regions representation

By preserving a linear structure of the constraints, we propose in the sequel a generic solution towards the binary variables reduction. To each of the regions in (2.39), we have

[9] There exists a finite M sufficiently large if and only if the polyhedra of type (2.17) are bounded. In the remaining of the manuscript, all the polyhedra of type (2.17) are assumed to be bounded for this reason.

associated in (2.40) a unique binary variable. Consequently, the total number of binary variables is N, the number of supporting hyperplanes (see (2.17)). However, a basic calculus shows that the minimum number of binary variables necessary to distinguish between these regions understood as logical alternatives is:

$$N_0 = \lceil log_2 N \rceil. \tag{2.43}$$

The value of N_0 is reached by observing that each of the N regions in (2.40) can be linked to a unique number. Taking these numbers successively (starting from zero), it follows that we need N_0 bits to codify them into a binary representation.

The question that arises is the following:
How to describe the regions in a linear formulation similar to (2.40) *through a reduced number of binary variables?*

We impose firstly that the binary expression appearing in the inequalities has to remain *linear* for computational advantages related to the optimization solvers. This structural constraint is equivalent with saying that any variable α_i should be described by a linear mapping in the form:

$$\alpha_i(\lambda) = a_0^i + \sum_{k=1}^{N_0} a_k^i \lambda_k, \tag{2.44}$$

where the symbols

$$(\lambda_1, \ldots, \lambda_{N_0}) \in \Lambda \triangleq \{0,1\}^{N_0}. \tag{2.45}$$

In the reduced space of Λ, we will arbitrarily associate a tuple:

$$\lambda^i \triangleq \left(\lambda_1^i \ldots \lambda_{N_0}^i\right) \tag{2.46}$$

to each region $\mathcal{R}_i^-$. Note that this association is not unique, and various possibilities can be considered: in the following, unless otherwise specified, the tuples will be appointed in lexicographical order.

The problem of finding a mapping in Λ, which describes region $\mathcal{R}_i^-$, reduces then to finding the coefficients $\left(a_0^i, a_1^i, \ldots, a_{N_0}^i\right)$ for which $\alpha_i(\lambda^i) = 0$ and $\alpha_i(\lambda^j) \geq 1$, $\forall j \neq i$ under mapping (2.44). This translates into the following conditions for any $\lambda^i, \lambda^j \in \{0,1\}^{N_0}$:

$$\begin{cases} a_0^i + \sum\limits_{k=1}^{N_0} a_k^i \lambda_k^i = 0, \\ a_0^i + \sum\limits_{k=1}^{N_0} a_k^i \lambda_k^j \geq 1, \quad \forall j \neq i, \end{cases} \tag{2.47}$$

with λ_k^i the k^{th} component of the tuple, λ^i, associated to $\mathcal{R}_i^-$.

Remark 2.9. Note that, in (2.47), the equality constraints for $j \neq i$ were relaxed to inequalities since the value of $M\alpha_i(\lambda^j)$ needs only to be sufficiently large (any $\alpha_i(\lambda^j) \geq 1$

being a feasible choice). Furthermore, the condition "≥ 1" can be relaxed to an arbitrary small positive constant by means of counterbalancing through an increase in constant M. □

Nothing is said a priori about the non-emptiness of the set described by (2.47). We need at least a point in the coefficient space $(a_0, a_1, \ldots, a_{N_0})$, which verifies conditions (2.47) in order to prove the non-emptiness. To this end, we present the following proposition:

Proposition 2.2. *[Prodan et al., 2012e] A mapping $\alpha_i(\lambda) : \{0,1\}^{N_0} \to \{0\} \cup [1, \infty[$ which verifies (2.47), is given by:*

$$\alpha_i(\lambda) = \sum_{k=1}^{N_0} t_k, \quad \text{where } t_k = \begin{cases} \lambda_k, & \text{if } \lambda_k^i = 0, \\ 1 - \lambda_k, & \text{if } \lambda_k^i = 1, \end{cases} \tag{2.48}$$

where λ_k denotes the k^{th} variable and λ_k^i its value for the tuple, λ^i, associated to region $\mathcal{R}_i^-$.

The coefficients $\left(a_0^i, \ldots, a_{N_0}^i\right)$ of the linear mapping (2.44) can be then obtained as:

$$a_0^i = \sum_{k=1}^{N_0} \lambda_k^i, \quad a_k^i = \begin{cases} 1, & \text{if } \lambda_k^i = 0 \\ -1, & \text{if } \lambda_k^i = 1 \end{cases}, \; k = 1, \ldots, N_0. \tag{2.49}$$

Proof. The claim is constructive; by introducing mapping (2.49) in (2.47), it can be verified by simple inspection that the conditions are fulfilled. □

Remark 2.10. Note that the problem of finding parameters α_i is independent of the actual shape of the polytope S. The coefficients obtained in (2.48) can be used for any topologically equivalent polytope (that is, with the same number of half-spaces). □

2.5.1.1 Prohibited tuples

By the choice of the cardinal N_0 as in (2.43), the number of tuples generated by the reduced set of binary variables (2.45) may be greater than the actual number of regions.

The tuples left unallocated will be labeled as *prohibited*, and additional inequalities will have to be added to the extended set of constraints (2.40) in order to effectively rander them infeasible to an optimization routine. These restrictions are justified by the fact that, under construction (2.49), an unallocated tuple will not enforce the verification of any of the constraints of (2.40) (see Remark 2.8). It then becomes evident that the single constraint of (2.41) has to be substituted by a set of constraints that implicitly make all the unallocated tuples infeasible.

The next corollary of Proposition 2.2 provides the means to construct an inequality which renders a tuple infeasible:

Corollary 2.1. *[Prodan et al., 2012e] Let there be a tuple* $\lambda^i \in \{0,1\}^{N_0}$. *The N-dimensional point that it describes, and exclusively this tuple is made infeasible by the combinations in* (2.46)*, with respect to the constraint:*

$$-\sum_{k=1}^{N_0} t_k^i \leq -\epsilon, \tag{2.50}$$

with t_k^i *defined as in Proposition 2.2 and* $\epsilon \in (0,1)$ *a scalar.* □

Proof. The left side of the inequality (2.50) will vanish only at tuple λ^i, and for the rest of the tuples in the discrete set $\{0,1\}^{N_0}$ will give values greater than or equal to 1. Thus, the only point made infeasible by inequality (2.50) is λ^i. □

The number of unallocated tuples may be significant, relationship in this sense being given by:

$$0 \leq N_{int} \leq 2^{\lceil log_2 N \rceil} - 2^{\lceil log_2 N \rceil - 1} - 1 = 2^{\lceil log_2 N \rceil - 1} - 1, \tag{2.51}$$

with the bound reached for the most unfavorable case of $N = 2^{\lceil log_2 N \rceil - 1} + 1$.

If we associate to each of the unallocated tuples an inequality as in Corollary 2.1, the efficiency of the associated optimization algorithm may still be improved. Indeed, such a feasible set representation can further be improved by noting (as previously mentioned) that the association between regions and tuples is arbitrary. One could then choose favorable associations which will permit more than one tuple to be removed through a single inequality. To this end, we present the next proposition:

Proposition 2.3. *[Prodan et al., 2012e] Let there be a collection of tuples* $\{\lambda^i\}_{i \in 1,\ldots,2^d} \in \{0,1\}^{N_0}$, *which completely spans a d-facet*[10] *of hypercube* $\{0,1\}^{N_0}$. *Let* $\mathcal{I}$ *be the set of the* $N_0 - d$ *indices, which retain a constant value over all the tuples* $\{\lambda^i\}_{i \in 1,\ldots,2^d}$ *composing the facet. Then, there exists one linear inequality*

$$-\sum_{k \in \mathcal{I}} t_k^* \leq -\epsilon, \tag{2.52}$$

which renders the tuples of the given facet (and only these ones) infeasible.

Variables t_k^* *and* ϵ *are taken as in Corollary 2.1 with* t_k^* *associated to* λ_k^*, *the common value of variable* λ_k *over the set of tuples* $\{\lambda^i\}_{i \in 1,\ldots,2^d}$. □

[10] Here, d denotes the degree of the facet, ranging from 0 for extreme points to $N_0 - 1$ for faces of the hypercube.

Proof. Geometrically, the tuples are extreme points on the hypercube $\{0,1\}^{N_0}$ and the inequalities we are dealing with are half-spaces, which separate the points of the hypercube. If a set of tuples completely spans a d-facet, it is always possible to isolate a half-space that separates the points of the d-facet from the rest of the hypercube. □

By a suitable association between feasible cells and tuples, we may label as unallocated the extreme points which compose entire facets on the hypercube $\{0,1\}^{N_0}$, which permits to apply Proposition 2.3 in order to obtain constraints (2.59).

Remark 2.11. By writing N_{int} as a sum of consecutive powers of 2 ($N_{int} = \sum_{i=0}^{\lceil log_2 N_{int} \rceil - 1} b_i 2^i$), an upper bound N_{hyp} for the number of inequalities (2.59) can be computed:

$$N_{hyp} = \sum_{i=0}^{\lceil log_2 N_{int} \rceil - 1} b_i \leq \lceil log_2 N_{int} \rceil, \tag{2.53}$$

where $b_i \in \{0,1\}$. □

Remark 2.12. Note that (2.53) offers an upper bound for the number of inequalities, but practically the minimal value can be improved depending on the method used for constructing the separating hyperplanes and of the partitioning of the tuples between the allocated and unallocated subsets. □

2.5.1.2 Illustrative example

As an illustration of the notions described in Section 2.5.1, we take the following square:

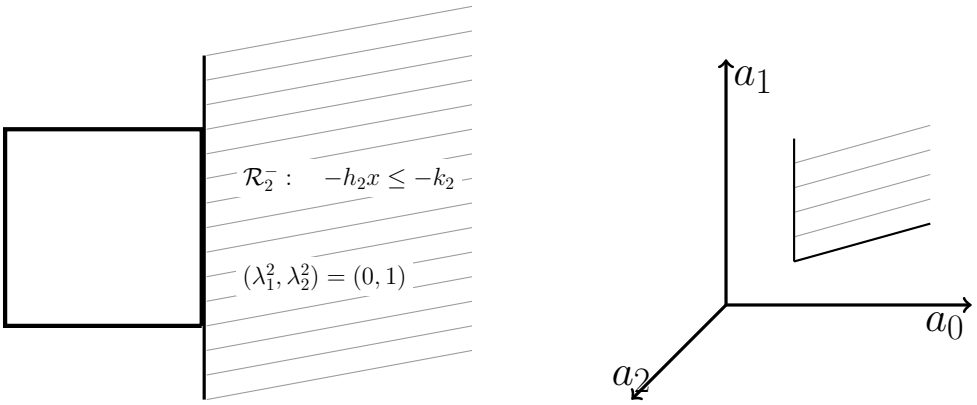
$$\begin{bmatrix} 0 & 1 \\ 0 & -1 \\ 1 & 0 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}. \tag{2.54}$$

As stated in this section, the number of binary variables (similar to the formulation (2.40)) is $N = 4$, equal to the number of half-spaces described in (2.54). The reduced number of variables will be $N_0 = \lceil log_2 4 \rceil = 2$, according to (2.43). Following the problem formulation (2.48), the variables α_i can be expressed, as in (2.44), by:

$$\alpha_i(\lambda) = a_0^i + a_1^i \lambda_1 + a_2^i \lambda_2.$$

We associate to each region a tuple of two values (λ_1, λ_2) in lexicographical order.

The case of the 2^{nd} half-space, associated to tuple $(\lambda_1^2, \lambda_2^2) = (0,1)$, is detailed in Figure 2.6(a). Using (2.47) we obtain, as depicted in Figure 2.6(b), the feasible set of the



(A) $R^-(\mathcal{H}_2)$ and its tuple (B) Feasible region for coefficients

FIGURE 2.6: Outer regions and their associated tuples

coefficients described by:

$$a_0^2 + a_2^2 = 0, \quad a_0^2 \geq 1, \quad a_1^2 \geq 1.$$

This represents a polytopic region in the coefficients space $(a_0, a_1, a_2) \in \mathbb{R}^3$ and, according to (2.48), the non-emptiness is assured by the existence of at least one feasible combination of coefficients leading to the mapping:

$$\alpha_2(\lambda) = 1 + \lambda_1 - \lambda_2.$$

This means that the region R_2^- is projected from

$$\begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq -1 + M(1 + \lambda_1 - \lambda_2),$$

by taking $(\lambda_1^2, \lambda_2^2) = (0, 1)$ (see Remark 2.8).

Furthermore, the same computations will be performed for the rest of the regions, resulting in an extended system of linear inequalities over mixed decision variables:

$$\begin{bmatrix} 0 & -1 \\ 0 & 1 \\ -1 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} -1 + M(\quad \lambda_1 + \lambda_2) \\ -1 + M(1 - \lambda_1 + \lambda_2) \\ -1 + M(1 + \lambda_1 - \lambda_2) \\ -1 + M(2 - \lambda_1 - \lambda_2) \end{bmatrix}.$$

As an exemplification of the considerations in Subsection 2.5.1.1, let there be a polytope with 5 hyperplanes. This means that the number of binary variables has to be $N_0 =$

$\lceil log_2 5 \rceil = 3$ and then, $N_{int} = 2^3 - 5 = 3$ tuples will remain unallocated; we choose these to be $(0, 0, 1)$, $(1, 0, 1)$ and $(1, 1, 1)$.

By applying Corollary 2.1, we observe in Figure 2.7 (a) the 3 inequalities that separate the unallocated tuples from the rest (for simplicity, in the rest of the subsection, we will use $\epsilon = 0.5$):

$$\begin{aligned} -(1 + \lambda_1 + \lambda_2 - \lambda_3) &\leq -0.5, \\ -(2 - \lambda_1 + \lambda_2 - \lambda_3) &\leq -0.5, \\ -(3 - \lambda_1 - \lambda_2 - \lambda_3) &\leq -0.5. \end{aligned}$$

We observe in Figure 2.7 (b) that the tuples are positioned onto 2 edges, and consequently, using Proposition 2.3, 2 inequalities suffice for separation:

$$\begin{aligned} -(1 \quad + \lambda_2 - \lambda_3) &\leq -0.5, \\ -(2 - \lambda_1 \quad - \lambda_3) &\leq -0.5. \end{aligned}$$

Lastly, recalling Remark 2.12, we note that, in this particular case, a single inequality (as seen in Figure 2.7 (c)), is enough for separating the unallocated tuples from the rest:

$$-(0.32\lambda_1 + 1.76\lambda_2 + 2.13\lambda_3) \leq -0.5.$$

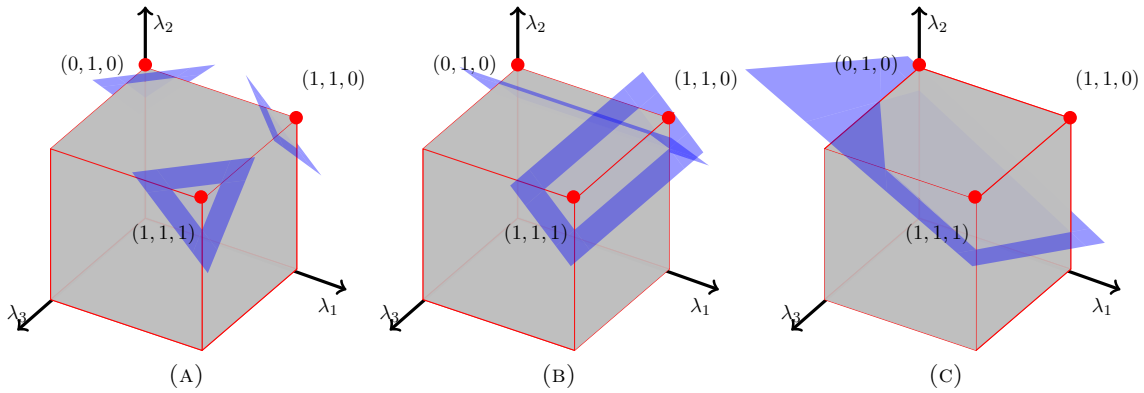


FIGURE 2.7: Exemplification of separating hyperplanes techniques

2.5.2 Description of the complement of a union of convex sets

In the previous section, the basic reduction method was applied for treatment of the complement of a convex set. A generalization of this important result case will be detailed in the following by considering the complement of a union of convex (bounded

polyhedral) sets $\mathbb{S} \triangleq \bigcup_l S_l$:

$$\mathcal{C}_X(\mathbb{S}) \triangleq cl(X \setminus \mathbb{S}), \tag{2.55}$$

with[11] $S_l \triangleq \bigcap_{k_l=1}^{\mathcal{K}_l} R^+(\mathcal{H}_{k_l})$ and $N \triangleq \sum_l \mathcal{K}_l$, with the limits of the sum such that the entire number of subregions defining $\mathbb{S}$ is covered.

This type of regions arises naturally in the context of obstacle/collision avoidance when there is more than a single object to be taken into account. In order to deal with the complement of a non-convex region in the context of mixed-integer techniques, we recall here and make use of the hyperplane arrangements definition, Definition 2.4 presented in Section 2.4.3.

Several computational aspects are of interest. The number of feasible cells, $\gamma(N)$, (in relation with the space dimension – d and the number of hyperplanes – N) is bounded by Buck's formula Buck [1943]:

$$\gamma(N) \leq \sum_{i=0}^{d} \binom{N}{i}, \tag{2.56}$$

with the equality satisfied if the hyperplanes are in general position[12] and $X = \mathbb{R}^n$.

An efficient algorithm for describing (2.29), based on reverse search that runs in $\mathcal{O}(N\gamma(N)lp(N,d))$ time and $\mathcal{O}(N,d)$ space, was presented in Avis and Fukuda [1996] and implemented in Ferrez and Fukuda [2002].

Note that there exists a subset $\{B_l\}_{l=1,\ldots,\gamma^b(N)}$ of *feasible polyhedral* cells from (2.29) (with $\gamma^b(N) \leq \gamma(N)$) which describes region (2.55):

$$\mathcal{C}_X(\mathbb{S}) = \bigcup_{l=1,\ldots,\gamma^b(N)} B_l, \tag{2.57}$$

such that, for any l, there exists a unique i for which, $B_l = A_i$ and $A_i \cap \mathbb{S} = \emptyset$.

In (2.40), a single binary variable was associated to a single inequality, but the mechanism can be applied similarly to more inequalities (e.g., the ones describing one of the cells of (2.57)). Thus, (2.29) can be described in an extended space of *state* and *auxiliary*

[11] The "+" superscript was chosen for the homogeneity of notation, equivalently one could have chosen any combination of signs in the half-space representation (2.19) in order to describe the polyhedral regions S_l.

[12] We call a hyperplane arrangement to be "in general position" whenever a variation in the position of the composing hyperplanes does not change the number of cells.

binary variables as follows:

$$\left.\begin{array}{c} \vdots \\ \sigma_l(1)h_1 x \leq \sigma_l(1)k_1 + M\alpha_l \\ \vdots \\ \sigma_l(N)h_N x \leq \sigma_l(N)k_N + M\alpha_l \\ \vdots \end{array}\right\} \quad B_l \tag{2.58}$$

and condition

$$\sum_{l=1}^{l=\gamma^b(N)} \alpha_l \leq \gamma^b(N) - 1, \tag{2.59}$$

which implies that at least a set of constraints will be verified.

Construction (2.58)–(2.59) will permit, through projection along the binary variables α_l (see (2.42)), to obtain any of the cells of the union (2.57).

Analogously to Section 2.5.1, we propose, in the following, the reduction of the number of binary variables by associating to each of the cells a unique tuple. The binary part will be computed following the constructive result in Proposition 2.2 and used accordingly in (2.58). Additional inequalities, that render infeasible the unallocated tuples, are introduced as in Proposition 2.3. A few remarks relating to the number of hyperplanes and their corresponding arrangement are in order:

Remark 2.13. The number of inequalities in (2.58) can be reduced by observing that not all the hyperplanes of $\mathbb{H}$ are active in a particular cell, and thus they can be discarded from the final representation. □

Remark 2.14. Note that a relaxation of the linear structure is allowed, when a nonlinear formulation involving products of binary variables can be employed and the hyperplane arrangements (2.29) can be represented as:

$$\begin{array}{c} \vdots \\ -h_i x \leq -k_i + M \cdot \displaystyle\prod_{\substack{l=1,\ldots,\gamma^b(N) \\ \sigma_l(i)='-'}} \alpha_l, \\ h_i x \leq \quad k_i + M \cdot \displaystyle\prod_{\substack{l=1,\ldots,\gamma^b(N) \\ \sigma_l(i)='+'}} \alpha_l, \\ \vdots \end{array} \tag{2.60}$$

for all sign tuples σ_l associated to cells B_l from covering (2.57). We have used the fact that the cells of (2.57) use the same half-spaces (up to a sign), and thus they can be

concatenated. The method presented in [Kobayashi and Imura, 2006] transforms an inequality with nonlinear binary components into a set of inequalities with linear binary components. However, this can be made only at the expense of introducing additional binary variables, which in the end gives a larger problem than the one presented in (2.58)–(2.59). □

2.5.2.1 Exemplification of hyperplane arrangements

Consider the following illustrative example depicted in Figure 2.8, where the complement of the union of two triangles ($\mathbb{S} = S_1 \cup S_2$) represents the feasible region. We take $\mathbb{H} = \{\mathcal{H}_i\}_{i=1:4}$ the collection of $N = 4$ hyperplanes (given as in (2.18)) which define S_1, S_2.

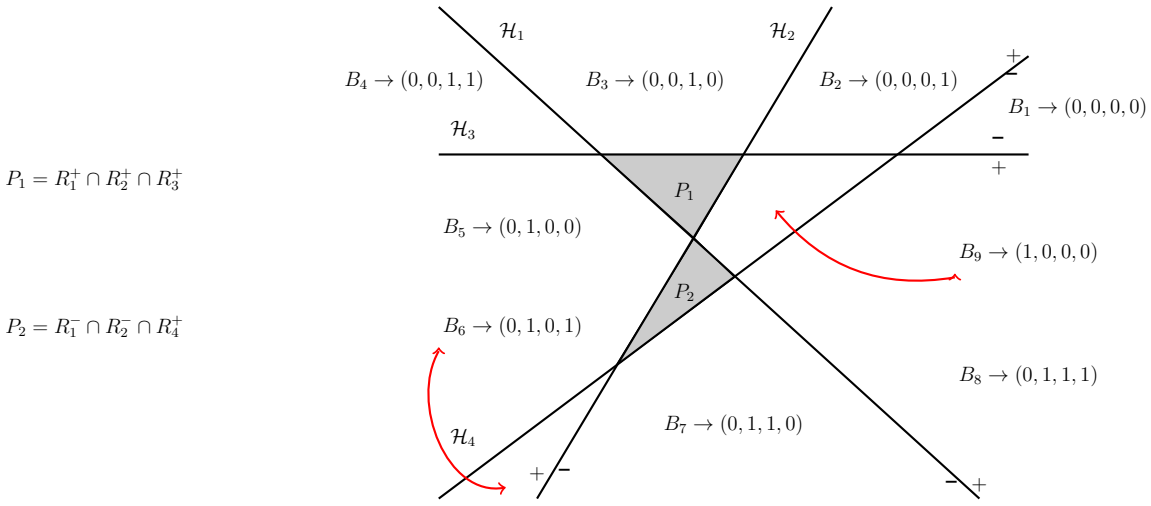


FIGURE 2.8: Exemplification of hyperplane arrangement

We observe that the bound given in (2.56) is reached, that is, we have 11 cells (obtained as in the arrangement (2.29)). From them, a total of 9, which we denote here as $B_1, \ldots, B_9$, describe the not convex region (2.55). To each of them, we associate a unique tuple from $\{0,1\}^{N_0}$ as seen in Figure 2.8 with $N_0 = \lceil log_2 9 \rceil = 4$.

As per Proposition 2.2 and (2.58), we are now able to write the set of inequalities (2.61).

$$
\begin{aligned}
&\left.\begin{aligned} -h_3x &\le -k_3 \\ h_4x &\le k_4 \end{aligned}\right\} + M\,(\qquad \lambda_1+\lambda_2+\lambda_3+\lambda_4) \Big\}\ B_1\\
&\left.\begin{aligned} -h_2x &\le -k_2 \\ -h_3x &\le -k_3 \\ h_4x &\le k_4 \end{aligned}\right\} + M\,(1+\lambda_1+\lambda_2+\lambda_3-\lambda_4) \Big\}\ B_2\\
&\left.\begin{aligned} h_1x &\le k_1 \\ h_2x &\le k_2 \\ -h_3x &\le -k_3 \end{aligned}\right\} + M\,(1+\lambda_1+\lambda_2-\lambda_3+\lambda_4) \Big\}\ B_3\\
&\left.\begin{aligned} -h_1x &\le -k_1 \\ -h_3x &\le -k_3 \end{aligned}\right\} + M\,(2+\lambda_1+\lambda_2-\lambda_3-\lambda_4) \Big\}\ B_4\\
&\left.\begin{aligned} -h_1x &\le -k_1 \\ -h_2x &\le -k_2 \\ h_3x &\le k_3 \\ h_4x &\le k_4 \end{aligned}\right\} + M\,(1+\lambda_1-\lambda_2+\lambda_3+\lambda_4) \Big\}\ B_5\\
&\left.\begin{aligned} h_2x &\le k_2 \\ -h_4x &\le -k_4 \end{aligned}\right\} + M\,(2+\lambda_1-\lambda_2+\lambda_3-\lambda_4) \Big\}\ B_6\\
&\left.\begin{aligned} h_1x &\le k_1 \\ h_2x &\le k_2 \\ -h_4x &\le -k_4 \end{aligned}\right\} + M\,(2+\lambda_1-\lambda_2-\lambda_3+\lambda_4) \Big\}\ B_7\\
&\left.\begin{aligned} h_1x &\le k_1 \\ h_3x &\le k_3 \\ -h_4x &\le -k_4 \end{aligned}\right\} + M\,(3+\lambda_1-\lambda_2-\lambda_3-\lambda_4) \Big\}\ B_8\\
&\left.\begin{aligned} h_1x &\le k_1 \\ h_2x &\le k_2 \\ h_3x &\le k_3 \\ h_4x &\le k_4 \end{aligned}\right\} + M\,(1+\lambda_1-\lambda_2+\lambda_3+\lambda_4) \Big\}\ B_9.
\end{aligned}
\tag{2.61}
$$

Note that, in the above set, we have simplified the description by cutting the redundant hyperplanes in a cell representation(e.g., for cell A_1, 2 hyperplanes suffice for a complete description).

Since only 9 tuples, from a total number of 16, are associated to cells, we need to add constraints to the problem such that the remaining 7 unallocated tuples will never be feasible. Using Corollary 2.1 we obtain:

$$
\begin{aligned}
&-(2-\lambda_1-\lambda_2+\lambda_3+\lambda_4)\leq -0.5,\\
&-(3-\lambda_1-\lambda_2-\lambda_3+\lambda_4)\leq -0.5,\\
&-(3-\lambda_1-\lambda_2+\lambda_3-\lambda_4)\leq -0.5,\\
&-(4-\lambda_1-\lambda_2-\lambda_3-\lambda_4)\leq -0.5,\\
&-(2-\lambda_1+\lambda_2-\lambda_3+\lambda_4)\leq -0.5,\\
&-(3-\lambda_1+\lambda_2-\lambda_3-\lambda_4)\leq -0.5,\\
&-(2-\lambda_1+\lambda_2+\lambda_3-\lambda_4)\leq -0.5.
\end{aligned}
\tag{2.62}
$$

We observe that for the 7 unallocated tuples, 4 of them, $(1,1,0,0)$, $(1,1,0,1)$, $(1,1,1,0)$ and $(1,1,1,1)$, form a 2-facet of the hypercube $\{0,1\}^4$. Tuples $(1,0,1,0)$ and $(1,0,1,1)$ form an edge and $(1,0,0,1)$ is on a vertex. We can now apply Proposition 2.3 and obtain the following constraints:

$$
\begin{aligned}
&-(2-\lambda_1-\lambda_2\qquad\quad)\leq -0.5,\\
&-(2-\lambda_1+\lambda_2-\lambda_3\quad)\leq -0.5,\\
&-(2-\lambda_1+\lambda_2+\lambda_3-\lambda_4)\leq -0.5.
\end{aligned}
\tag{2.63}
$$

Note that we were able to diminish the number of inequalities from 7 in (2.62) to only 3 in (2.63): the first 4 constraints of (2.62) are replaced by the 1^{st} constraint of (2.63). The same holds for the next 2 that correspond to the 2^{nd}, and for the last that is identical with the 3^{rd}.

2.5.3 Refinements for the complement of a union of convex sets

As presented in [Stoican et al., 2011b], palliatives for reducing the computational load exist but ultimately, the computation time is in the worst case scenario exponentially dependent on the number of binary variables, which in turn depends on the number of cells of the hyperplane arrangements (see (2.56)). We conclude then, that the problem becomes prohibitive for a relatively small number of polyhedra in $\mathbb{S}$, and that any reduction in the number of cells is worthwhile and should be pursued.

This can be accomplished in two complementary ways. Firstly, we note that bound (2.56) is reached for a given number of hyperplanes if and only if they are in general position. As such, particular classes of polyhedra may somewhat reduce the actual number of cells in arrangement (2.29) and, consequently, the number of auxiliary binary variables. In increasing order of their versatility, we may mention hypercubes, orthotopes, parallelotopes and zonotopes as classes of interest (for a computation of the number of cells (see, [Zaslavsky, 1975])).

The other direction, which we chose to pursue in the rest of this section, is reducing the number of cells that describe (2.55).

In Stoican et al. [2011b], we have proposed a hybrid scheme which permits to express (2.55) as a union of the cells of (2.57) which intersect $\mathbb{S}^\circ$, and of the regions (in the sense of (2.19)) which describe $\mathcal{C}(\mathbb{S}^\circ)$, where $\mathbb{S}^\circ$ denotes the convex hull of $\mathbb{S}$.

Alternatively, the merging of adjacent cells into possibly overlapping regions, which describe (2.55), is discussed in Stoican et al. [2011c]. This results in a reduced representation, both in number of cells and of interdicting constraints. In the next subsection, we detail the merging techniques used and show how the complexity of the problem is reduced.

2.5.4 Cell merging

Recall that any of the cells of (2.57) is described by a unique sign tuple $(B_l \leftrightarrow \sigma_l)$. As such, we obtain that the cells are disjunct and cover the entire feasible space. For our purposes, we are satisfied with any collection of regions not necessarily disjoint which covers the feasible space. In this context, we may ask if it is not possible to merge the existing cells of (2.57) into a reduced number of regions which will still cover region (2.55). Note that by reducing the number of regions, the number of binary auxiliary variables will also decrease substantially.

We can formally represent the problem by requiring the existence of a collection of regions,

$$\mathcal{C}_X(\mathbb{S}) = \bigcup_{k=1,\dots,\gamma^c(N)} C_k, \tag{2.64}$$

which verifies next conditions:

- the new polyhedra are formed as unions of the old ones (i.e., for any k there exists a set I_k which selects indices from $1,\dots,\gamma^b(N)$ such that $C_k = \bigcup_{i\in I_k} B_i$),
- the union is minimal, that is, the number $\gamma^c(N)$ of regions is minimal.

Existing merging algorithms are usually computationally expensive, but here we can simplify the problem by noting two properties of the cells in (2.57):

- the sign tuples σ_l describe an adjacency graph since any two cells whose sign tuples differ at only one position are neighbors,
- the union of any two adjacent cells is a polyhedra.

In order to construct (2.64), we may use merging algorithms (see, for example, [Geyer et al., 2004], which adapts a "branch and bound" algorithm to merge cells of a hyperplane arrangement), or we can pose the problem in the boolean algebra framework. The merging problem of regions from (2.57) is functionally identical to the minimization of a boolean function given in the "sum of products" form. A cell describing the (in)feasible region (2.55) corresponds to a "1" ("0") value in the truth table at the position determined by its associated sign tuple, whereas infeasible sign tuples correspond to "don't care" values. It is then straightforward to apply minimization algorithms (Karnaugh maps, the Quine-McCluskey algorithm or the Espresso heuristic logic minimizer) in order to obtain boolean minterms who describe the merged cells of (2.64). We note that a similar approach was proposesd in [Geyer et al., 2008] in order to deal with polyhedral piecewise affine systems.

Remark 2.15. Note that a region C_k is described by at most $N - f$ hyperplanes, where f denotes the number of indices in the sign tuples which flip the sign. It makes sense then to, not only reduce the number of regions, but also to maximize the number of cells that go into the description of a region from (2.64). □

In Algorithm 2.1 we sketch the notions presented in this section.

Algorithm 2.1: Scheme for representing $\mathcal{C}(\mathbb{S})$

Input: $\mathbb{S}$

1 obtain the cell arrangement as in (2.29) for $\mathbb{H}$;
2 obtain the feasible cells (2.57) and merge them in representation (2.64);
3 get the number $\gamma^c(N)$ of feasible regions and the number N_0 of auxiliary binary variables;
4 partition the tuples of $\{0,1\}^{N_0}$ such that Proposition 2.3 can be efficiently applied;
5 create the extended polyhedron (2.58) and add the constraints (2.59)

Note that the steps 1 and 2 are the most computationally expensive. For the first step, an efficient algorithm for describing (2.29), based on reverse search that runs in $\mathcal{O}(N\gamma(N)lp(N,d))$ time and $\mathcal{O}(N,d)$ space, was detailed in [Avis and Fukuda, 1996] and implemented in [Ferrez and Fukuda, 2002]. For the second step, boolean algebra techniques have been used: for a small number of hyperplanes, exact methods like Karnaugh maps prove to be effective. For higher numbers, the heuristic Espresso logic minimizer can be used. The latter provides solutions very close to the optimum while being more efficient and reducing memory usage and computation time by several orders of magnitude relative to classical methods (see, [Rudell and Sangiovanni-Vincentelli, 1985]).

2.5.4.1 Exemplification of hyperplane arrangements with cell merging

We revisit here the example provided in Subsection 2.5.2.1 and apply the results presented in Subsection 2.5.4 in order to show the improvements.

For this simple case we apply, as seen in Figure 2.9, a Karnaugh diagram and obtain that the feasible region (2.55) is expressed by a union as in (2.58).

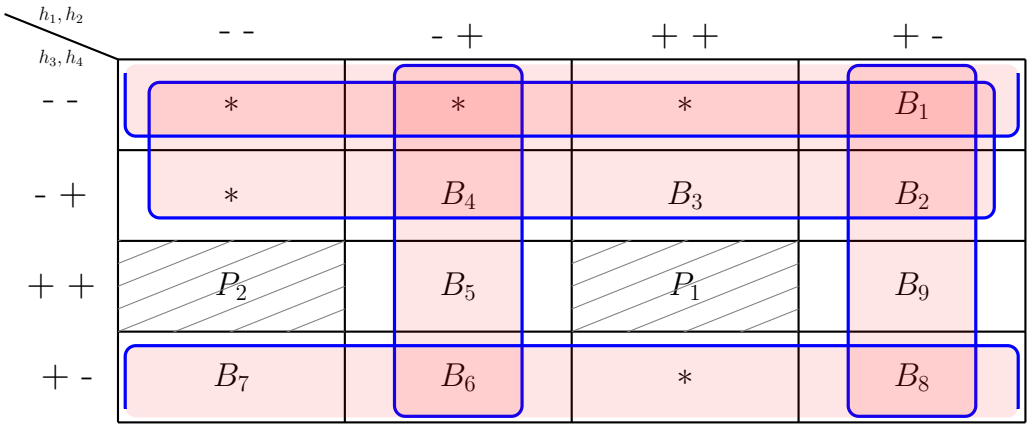


FIGURE 2.9: Karnaugh diagram for obaining the reduced cell representation

As seen from the Karnaugh diagram from Figure 2.9, we obtain 4 overlapping regions: $C_1 = B_1 \cup B_2 \cup B_3 \cup B_4$, $C_2 = B_4 \cup B_5 \cup B_6$, $C_3 = B_6 \cup B_7 \cup B_8 \cup B_1$ and $C_4 = B_8 \cup B_9 \cup B_1 \cup B_2$, which we depict in Figure 2.10. Consequently, we note that $N_0 = 2$ auxiliary binary variables suffice in coding the regions. As for Proposition 2.2 and (2.58), we are now able to write the following set of inequalities (we attach to each of the regions a tuple in lexicographical order):

$$\begin{aligned}
&-h_3 x \le -k_3 + M\,(\qquad\ \lambda_1 + \lambda_2)\Big\} && C_1,\\
&\left.\begin{matrix} -h_1 x \le -k_1 \\ -h_4 x \le -k_4 \end{matrix}\right\} + M\,(1 + \lambda_1 - \lambda_2) && C_2,\\
&-h_4 x \le -k_4 + M\,(1 - \lambda_1 + \lambda_2)\Big\} && C_3,\\
&\left.\begin{matrix} h_1 x \le\ \ k_1 \\ -h_2 x \le -k_2 \end{matrix}\right\} + M\,(2 - \lambda_1 - \lambda_2) && C_4.
\end{aligned} \tag{2.65}$$

Note that, in addition to reducing the number of regions in (2.65) comparative with (2.61), we also have reduced the number of hyperplanes appearing in the region's half-space representation (see Remark 2.11).

2.5.5 Numerical considerations

In this section, we will test the computation time improvements for our approach versus the standard technique encountered in the literature. As previously mentioned, a MIP

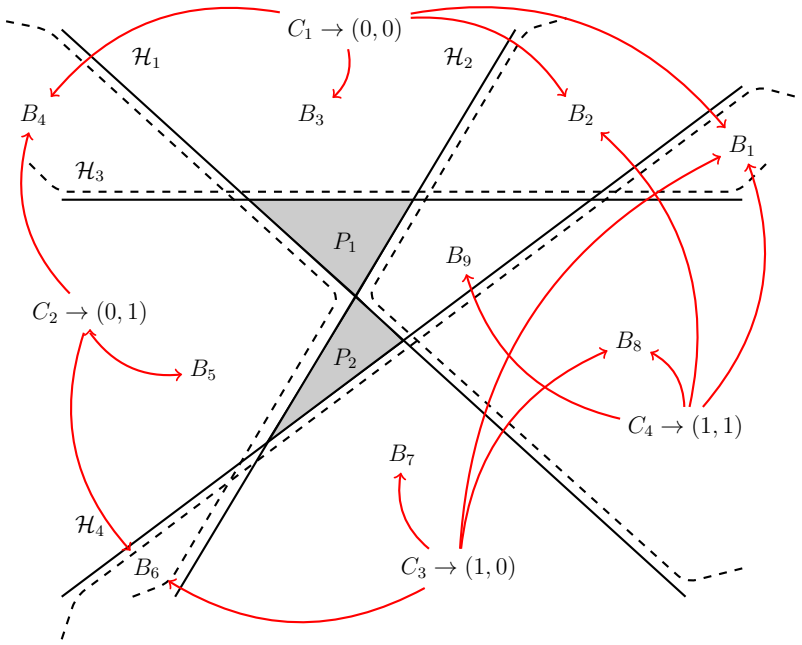


FIGURE 2.10: Exemplification of hyperplane arrangement with merged regions

problem is NP-hard in the number of binary variables (due to the fact that the algorithms implement "branch-and-bound" techniques and as such, in the worst case scenario, they need to iterate through all the branches defined by the binary search tree variables). Therefore, even a small reduction will render sensible improvements.

The complexity of the MIP optimization algorithm with constraints in the classical form (2.40)–(2.41) will be of the order of $\mathcal{O}(2^N \cdot p(N+1,d))$, where $p(n,d)$ denotes either $lp(n,d)$ or $qp(n,d)$, as it is required. Using the alternative formulation proposed in Section 2.5.1, we obtain the complexity as:

$$\mathcal{O}(2^{\lceil log_2 N \rceil} \cdot p(N + \lceil log_2 N \rceil - 1), d) = \mathcal{O}(N \cdot p(N,d)). \tag{2.66}$$

We consider the worst case scenario for the mixed-integer problem, that is, the optimization algorithm will need to pass through each of the possible combinations provided by the binary variables. In the case of formulation (2.40)–(2.41), this leads to a NP complexity in the number of lp/qp problems to be solved. On the other hand, the formulation shown in (2.66), through the reduction of binary variables, provides a polynomial complexity.

To illustrate these computational gains, we will compare the times of execution for both schemes as follows: the computational time will be measured and averaged for 10 samples of 2D polytopes with the same number of support hyperplanes; furthermore,

no. of hyperplanes	5	10	15	20	25	50	100
classical	9.91	64.06	91.74	511.47	306.04	···	···
enhanced	1.14	0.81	0.59	4.84	4.18	3.66	2.94

TABLE 2.1: Numerical values for the solution time of an MI optimization problem under classical and enhanced methods.

the procedure will be iterated by changing the number of hyperplanes from 4 to 25. The results are depicted in Figure 2.11 on a semilogaritmic scale and, as it can be seen, there are significant improvements. In fact, the differences may be even more pronounced since, under default settings, the MI algorithm over the classical method stopped computing the optimum value after a maximum number of iterations was reached (the MI algorithm used was the one described in [Bemporad and Morari, 1999]).

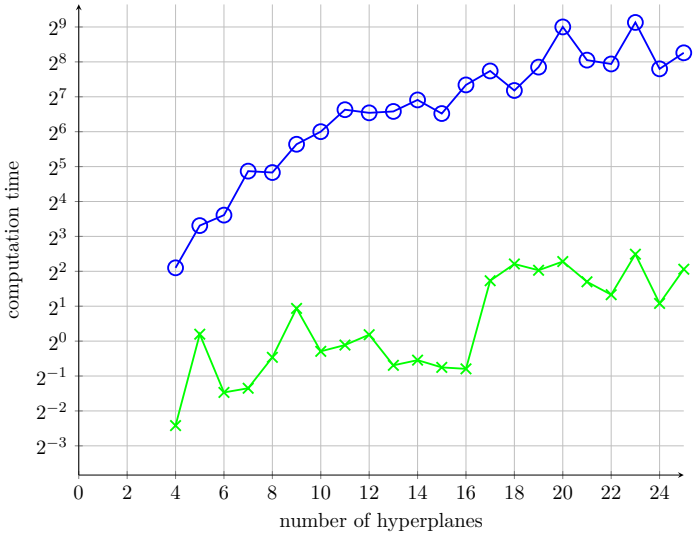


FIGURE 2.11: Comparative test for computation time for classical and enhanced method – time axis in logarithmic scale

Similar results are shown in Table 2.1, where we observe the evident improvement of our method relative to the classical technique.

In Section 2.5.2, a method for describing in the MI formalism of the complement of a possibly not connected union of polytopes was presented. The main drawback is that in both classical and reduced formulation, the problem depends on the number of cells. Supposing that the hyperplanes from the hyperplane arrangement (2.57) be in random position, we obtain, for formulation (2.58)–(2.59), a complexity of order

$$\mathcal{O}(2^{\gamma^b(N)} \cdot p(N \cdot N^d + 1), d), \tag{2.67}$$

which can be further reduced, using the techniques from Section 2.5.1, to

$$\mathcal{O}(2^{\lceil log_2 \gamma^b(N) \rceil} \cdot p(N \cdot N^d + 1, d)) = \mathcal{O}(\gamma^b(N) \cdot p(N^{d+1}, d)). \tag{2.68}$$

Again, we observe that the mixed-integer problem becomes polynomial in the number of branches: in the worst case, we have γ^b subproblems to solve, and $\gamma^b \leq \gamma$ which in turn is given by the polynomial expression (2.56). However, the problem is still challenging due to the number of cells (see (2.56)). By reducing the number of cells as in Section 2.5.3, it is possible to significantly reduce the computation time. For exemplification, take the example depicted in Figures 2.8 and 2.10. We observe that in this particular case, we were able to reduce the representation from 9 cells to only 4. Presumably, for a higher number of hyperplanes, the gain will be even more pronounced.

2.6 Concluding remarks

In this chapter we introduced a global optimization-based approach combined with set-theoretic methods in order to deal with the problems appearing in the control of multi-agent dynamical systems. Note that this approach can always be considered as a solution for basic control problems with collision avoidance constraints and trajectory planning. However, there are a lot of challenging issues that arise in the context of cooperative systems. In the forthcoming chapters, we will show how the proposed approach can be adapted in order to solve them.

For example, when dealing with basic tasks as trajectory tracking of multiple agents, we will use differential flatness tools. Also, for having a robust approach we will construct invariant sets which characterize the dynamics of the agents. Furthermore, these invariant sets will represent safety regions around agents in order to avoid collisions for any values of the bounded disturbances affecting the agents. A particular contribution is envisaged by the use of polyhedral norm in order to construct the so called polyhedral function and sum function, which, in the next chapter, will represent the basis for the construction of penalty functions.

Since in our particular context the non-convex constraints are ubiquitous, the most part of this chapter presents a novel approach based on Mixed-Integer Programming (MIP), for describing in a linear form non-convex, non-connected feasible regions. More precisely, we were able to transform the feasible region into a polyhedron in an augmented space (state and auxiliary binary variables) through the use of hyperplane arrangements. Furthermore, using cell merging techniques we minimized the number of cells describing the feasible region, thus bringing a notable improvement in the constraints handling for mixed-integer optimization problems. These numerical improvements will give an extra weight on the approach chosen in the next chapters for solving obstacle and collision avoidance control problems in the context of multi-agent systems.

Chapter 3

Multi-agent formation control

Formation control has become one of the well-known problems in multi-agent dynamical systems. Compared with a single agent, many advantages of a group of agents working together have been shown in various applications involving the control of cooperative systems [Grundel et al., 2007], such as object transportation [Burmeister et al., 1997], [Negenborn et al., 2008], mobile sensor networks [Olfati-Saber, 2006], cooperative classification and surveillance where the sensor assets are limited [Pavón et al., 2007]. Here, the agent formations allow each member of the team to concentrate its sensors across a specified region of the workspace, while their neighboring members cover the rest of the workspace. Furthermore, in applications like search and rescue, coverage tasks and security patrols, the agents direct their visual and radar search responsibilities depending on their positions in the formation [Balch and Arkin, 1998]. Other applications include coordinated ocean platform control for a mobile offshore base [Girard et al., 2001]. In this case, the homogeneous modules forming the offshore base must be able to perform long-term station keeping at sea, in the presence of waves, winds and currents. Consequently, the independent modules have to be controlled in order to be maintained aligned (i.e., secure their convergence to a tight formation).

The characterization and the convergence towards a formation represent classical issues for the control of cooperative systems. Moreover, the specific problem of maintaining a formation becomes even more challenging if one needs to ensure that all the agents avoid collisions inside the group and/or with obstacles [Richards and How, 2002]. The primary challenge in solving collision avoidance problems is often a result of the non-convex constraints appearing in the real-time optimization problem related to the control (see, for more details Chapter 2 of the present manuscript).

Needless to say there exists extensive research in the area of multi-agent systems and more specifically, on the directions of formation control and collision avoidance (see also the Introduction chapter of the present manuscript). The contribution of the present manuscript on these directions is towards an understanding of the geometrical properties

and a novel (to the best of the authors' knowledge) synthesis framework which exploits the same geometrical structure of the issues discussed here. The formation design and the collision avoidance conditions are casted as geometrical problems and optimization-based procedures are developed to solve them. Moreover, we have obtained considerable advances in this direction by using efficiently MIP techniques (in order to derive a coherent description of the feasible region in the solution space) and stability properties (in order to analyze the uniqueness and existence of formation configurations). This results were first presented in [Prodan et al., 2011b], [Prodan et al., 2012c], [Prodan et al., 2012d].

In the present chapter we concentrate on the optimization-based control of multiple agents having independent dynamics while achieving a global objective, such as a tight formation with desired specifications and collision free behavior. For reducing the computation time we use the "nominal" behavior of the agents and consider safety regions around them to compensate for the effects of the disturbances affecting the "real" systems. Further, these regions are defined within the theory of invariant sets in order to avoid recomputations during the real-time functioning. Note that this choice guarantees also a degree of robustness despite the fact that the real-time control is performed using nominal prediction models.

The proposed approach is to decompose the formation control problem in two separate problems:
• The "off-line" definition of the ideal configuration. A minimal configuration is determined with respect to a given cost function under the constraints imposed by the safety regions. A particular contribution here is the introduction of an additional fixed point constraint (i.e., the target positions are also equilibrium points for the considered dynamics) in order to ensure convergence towards the predefined configuration.
• In real-time, a receding horizon optimization combined with task assignment relative to the minimal configuration will be employed.
The real-time control is designed based on the following two-stage procedure:

1. Determine "who-goes-where" in the formation. This is equivalent with solving a standard assignment problem. As detailed in the Introduction chapter of the present manuscript, this problem is not new in the field of combinatorial optimization [Hallefjord Kurt, 1993]. However, in the multi-agent control framework, the optimal assignment in real-time brings a novel (dynamical) dimension in terms of time-varying target.

2. Solve a mixed-integer optimization problem according to the target geometry of the formation and the associated safety regions in order to obtain the effective control actions for the individual agents.

Finally, these two separate problems are embedded within a receding horizon optimization-based control, leading to an optimization problem for driving the group of agents to a specified formation with associated target locations.

In the present chapter, we propose to tackle the above construction from a *centralized* and then, *distributed* implementation point of view. With all the improvements considered in the enhancements of MIP formulation, the centralized multi-agent problem is still difficult to solve, even more so when a predictive control scheme is applied. Consequently, in the second part of the chapter we extend the MIP solution in the context of distributed predictive control. Providing the same geometric description of the feasible region, a distributed MPC approach will be presented, upon the principles in [Dunbar and Murray, 2006], [Scattolini, 2009]. More precisely, we consider neighborhoods which partition the agents into groups and give a MIP description of the feasible region in which they stand. With an adequate communication between the resulting groups of agents, we are able to respect performance and stability constraints while in the same time we greatly reduce the computational load.

In the last part of the chapter, we introduce a *decentralized* control strategy which is a combination of MPC and Potential Field approach. Arguably, the potential field-based methods offer a less complex implementation with a smaller computational effort. However, in this case the constraints are no longer "hard" but rather "soft", in the sense that we impose penalties in crossing them through the cost function. We emphasize that our interest (and the main contribution here) is mainly in taking into account the shape of safety region of an agent navigating by the obstacles and/or other moving entities. The constructions (2.26), (2.30) proposed in the previous sections (see, Section 2.4.2 and Section 2.4.3, respecively, from Chapter 2) based on the class of (symmetrical) piecewise linear functionals defined using a specific shape of a polyhedral set as in (2.4.1) will be employed as various potential or navigating functions along the classical principles existing in the literature. Furthermore, the solution we give is decentralized, the steering policy for each agent is based only on local state information from its nearest neighbors.

Throughout the rest of the chapter we will detail the theoretical results with illustrative examples and simulations meant to highlight the ideas provided here. Lastly, we will draw the conclusions and end with a comparison of the a priori discussed methods.

3.1 System description

Consider a set of N_a linear systems (vehicles, pedestrians or agents in a general form) which model the behavior of individual agents.

Let us define

$$\mathcal{I} \triangleq \{1, \ldots, N_a\}, \tag{3.1}$$

as the collection of all agents indices.

The i^{th} system is described by the following discrete LTI dynamics affected by additive disturbances (which represents in fact, a simplified form of the model (2.9) described in

Section 2.2, Chapter 2):

$$x_{d,i}(k+1) = A_i x_{d,i}(k) + B_i u_{d,i}(k) + w_i(k), \quad i \in \mathcal{I}, \tag{3.2}$$

where $x_{d,i}(k) \in \mathbb{R}^n$ are the state variables, $u_{d,i}(k) \in \mathbb{R}^m$ is the control input and $w_i(k) \in \mathbb{R}^n$ represents a bounded disturbance for the agent i. Henceforth, we assume the following:

1. The pair (A_i, B_i) is stabilizable, with $A_i \in \mathbb{R}^{n\times n}$, $B_i \in \mathbb{R}^{n\times m}$.
2. The disturbance w_i is bounded, i.e. $w_i \in \mathcal{W}_i$, where $\mathcal{W}_i \subset \mathbb{R}^n$ is a convex and compact set containing the origin.

Theoretically, the dynamics (3.2) suffices to address any typical multi-agent control problem (e.g., formation stability, trajectory tracking and so forth). However, the presence of additive noises makes the numerical computation difficult and severely limits the practical implementability. This is particularly true for centralized schemes where the computations are to be made into an extended state-space generated as a cartesian product of the individual states of the N_a agents.

The control design principle followed here is based on the ideas in [Mayne et al., 2005]. As a first step, we consider the nominal systems associated to (3.2):

$$x_i(k+1) = A_i x_i(k) + B_i u_i(k), \quad i \in \mathcal{I}. \tag{3.3}$$

The real system described by (3.2) will be used in oder to describe the characteristics of the minimal group formation at the "off-line" stage. The control defines the target positions and the shape of safety region around them. The nominal system (3.3) is used "on-line" for the optimization-based real-time control. This will provide a computational efficient nominal trajectory. The state of the real system is then expected to reside in a tube around this nominal trajectory [Raković et al., 2005], [Mayne et al., 2005].

Furthermore, the existence of a stabilizable pair (A_i, B_i) implies the existence of an optimal control law for each agent i, $K_i \in \mathbb{R}^{n\times m}$ such that the matrices $A_i + B_i K_i$ are stable, where the controller K_i, $i \in \mathcal{I}$ is constructed either by a LQ design using the solution of the discrete algebraic Riccati equation or alternatively by pole placement technique. In both cases the existence of a quadratic Lyapunov function is supposed to account for the infinite-time cost function:

$$V(x_i) = x_i^T P x_i, \tag{3.4}$$

with $P = P^T \succ 0$.

Taking into account the above remarks, the control laws associated to dynamics (3.2) and (3.3), respectively, can be linked through the relation:

$$u_{d,i}(k) = u_i(k) + K_i(x_{d,i}(k) - x_i(k)), \tag{3.5}$$

such that the tracking error of the i^{th} nominal system (3.3), which is described by $z_i(k) \triangleq x_{d,i}(k) - x_i(k)$, has the dynamics:

$$z_i(k+1) = (A_i + B_i K_i) z_i(k) + w_i(k), \quad i \in \mathcal{I}. \tag{3.6}$$

Remark 3.1. The existence of a stabilizing gain K_i in (3.5)–(3.6) is assured by the existence of the Lyapunov function (3.4). Needless to say, if (3.4) does not exist, the tracking error (3.6) is not bounded and thus the tube-MPC approach is not applicable. □

Since we have assumed matrix K_i to stabilize the dynamics (3.3), it follows that the tracking error dynamics (3.6) are also stabilizable. Consequently, the resulting trajectories will converge towards a RPI set (see Definition 2.6 in Section 2.4.4, Chapter 2) S_i, which can be determined such that the following expression holds:

$$z_i(k) = x_{d,i}(k) - x_i(k) \in S_i, \quad i \in \mathcal{I}, \tag{3.7}$$

as long as[1] $z_i(0) \in S_i$ for any $k \geq 0$.

With these basic remarks we observe that it is sufficient to adjust the trajectory based on the nominal system (3.3), which does not depend on disturbance. Practically, the existence of this invariant set guarantees that the real system (3.2) resides in a tube

$$x_{d,i}(k) \in x_i(k) \oplus S_i, \quad i \in \mathcal{I}, \tag{3.8}$$

along the reference trajectory $x_i(k)$ for all $u_i(k)$, with $i \in \mathcal{I}$.

As mention in Section 2.4.4, Chapter 2 the set S_i in (3.8) can be constructed as an invariant approximation of the minimal robustly invariant set (mRPI) (see Definition 2.8). In most of our constructions we will use an ultimate bound construction, due to its low computational demands (the reader is referred to Theorem 2.2 for the construction of the RPI set using ultimate bounds).

Hence, the above manipulations allow us to consider the nominal system in the subsequent optimization problems and thus, minimize the necessary numerical computations. Moreover, for the collision avoidance problem, the nominal dynamics (3.3) will be used, but it is required that the *safety region of the real agents* do not intersect. Their true

[1]The assumption that the tracking error starts inside the set is made for simplification reasons. As long as the set is contractive, after a finite number of steps, any trajectory starting "outside" will enter inside the set.

position is unknown (due to the disturbance affecting them), but a collision avoidance policy will be based on the fact that an agent i will always reside in a region:

$$\mathcal{S}_i \triangleq x_i(k) \oplus \Omega_{UB,i}, \tag{3.9}$$

where $\Omega_{UB,i}$ is the RPI set under the dynamics (3.6), as in Theorem 2.2, with $i \in \mathcal{I}$.

Remark 3.2. For the ease of the computation, the agents are considered as unidimensional points in the position space. Even if they are characterized by a nonempty region $R_i \subset \mathbb{R}^n$, one can define the set in (3.9) as $\tilde{\mathcal{S}}_i \triangleq \mathcal{S}_i \oplus R_i$. □

Exemplification for construction of RPI sets

For illustration purposes let us consider a set of N_a heterogeneous agents in two spatial dimensions with the dynamics described by [2] $\dot{\xi}_i = A_i\xi_i + B_iu_i + w_i$:

$$A_i = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{\mu_i}{m_i} & 0 \\ 0 & 0 & 0 & -\frac{\mu_i}{m_i} \end{bmatrix}, \; B_i = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{m_i} & 0 \\ 0 & \frac{1}{m_i} \end{bmatrix} \tag{3.10}$$

affected by the additive disturbances:

$$|w_i| < [0.5 \; 0.3 \; 0.5 \; 0.2]^T, \tag{3.11}$$

where $\dot{\xi}_i = [x_i \; y_i \; v_{x,i} \; v_{y,i}]^T$, $u_i = [u_{x,i} \; u_{y,i}]^T$ are the state and the input of each system. The components of the state are: the position (x_i, y_i) and the velocity $(v_{x,i}, v_{y,i})$ of the i^{th} agent, $i \in \mathcal{I}$. The parameters m_i, μ_i are the mass of the i^{th} agent and the damping factor, respectively.

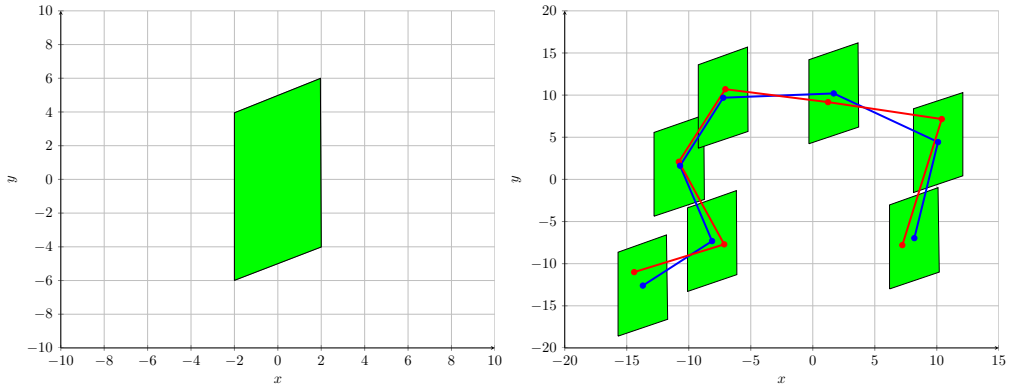
The dynamics (3.10) are stated in the continuous domain to better emphasize the structure of the problem but, further on, we will use a discretization of the system using a first order discretization method leading to:

$$A_{d,i} = \begin{bmatrix} 1 & 0 & 0.09 & 0 \\ 0 & 1 & 0 & 0.09 \\ 0 & 0 & 0.96 & 0 \\ 0 & 0 & 0 & 0.96 \end{bmatrix}, \; B_{d,i} = \begin{bmatrix} 0.001 & 0 \\ 0 & 0.001 \\ 0.016 & 0 \\ 0 & 0.016 \end{bmatrix} \tag{3.12}$$

obtained with a sampling time $h = 0.1$ and $m = 60$, $\mu = 3$.

For the sake of illustration, we construct the RPI set (i.e., the safety region around an agent) for a single agent affected by disturbances. Using pole placement methods we

[2] In the simulations results presented throughout the present chapter the agents are described by (3.10), which is a model used frequently in pedestrian flow applications [Fang et al., 2010].



(A) Projection of the RPI set on the position subspace.

(B) Trajectories of the "real" (in red) and "nominal" (in blue) system.

FIGURE 3.1: Exemplification for construction of RPI sets.

derive the feedback gain matrix:

$$K_1 = \begin{bmatrix} 3 & 2.2 & -2 & 0.1 \\ 0.1 & -1.3 & 0.5 & 1 \end{bmatrix}.$$

The RPI set S_1 is obtained using the techniques presented in Section 2.4.4, Chapter 2 and the projection on the position subspace is depicted in Figure 3.1 (a). For this system a nominal trajectory (3.3) (in blue) is constructed and one observes in Figure 3.1 (b) that any trajectory of system (3.2) affected by disturbance will verify relation (3.8) (i.e., resides in a tube described by the RPI set S_1).

To summarize, once a robustly positive invariant set (3.9) around each agent (3.2) is defined, one can deal with the collision avoidance problem within the multi-agent formation control using these invariant sets as a safety regions around the nominal position of the agents.

3.2 Collision avoidance formulation

A typical multi-agent problem is represented by the minimization of some cost problem with constraints. As stated before, the original formulation, with dynamics (3.2) is not optimal since it requires to take into account the bounded disturbances affecting the dynamics. Hereafter, we will use the "nominal" dynamics (3.3) and we will analyze how conditions on the "real" dynamics (3.2) are transposed to the "nominal" dynamics case.

For further use, let us define

$$\mathcal{I}_o \triangleq \{1, \ldots, N_o\}, \tag{3.13}$$

as the collection of all fixed obstacles indices. Furthermore, we describe the collection of obstacles through a union of polyhedra $\{\mathcal{O}\}_{o=1:N_o}$, with N_o denoting the number of polyhedra $\mathcal{O}_o$.

Considering the "real" dynamics (3.2), the collision/obstacle avoidance conditions translates into[3]:

$$\{x_{d,i}\} \cap \{x_{d,j}\} = \emptyset, \qquad \forall i, j \in \mathcal{I},\ i \neq j, \tag{3.14a}$$
$$\{x_{d,i}\} \cap \mathcal{O}_o = \emptyset, \qquad \forall i \in \mathcal{I},\ o \in \mathcal{I}_o, \tag{3.14b}$$

with $\mathcal{I}$ and $\mathcal{I}_o$ defined as in (3.1) and (3.13), respectively. Furthermore, using the notation from (3.3) and assuming that the conditions validating relation (3.8) are verified, we reach the equivalent formulation of the collision avoidance:

$$(\{x_i\} \oplus \mathcal{S}_i) \cap (\{x_j\} \oplus \mathcal{S}_j) = \emptyset, \qquad \forall i, j \in \mathcal{I},\ i \neq j, \tag{3.15a}$$
$$(\{x_i\} \oplus \mathcal{S}_i) \cap \mathcal{O}_o = \emptyset, \qquad \forall i \in \mathcal{I},\ o \in \mathcal{I}_o. \tag{3.15b}$$

These conditions take explicitly (through the use of the sets $\mathcal{S}_i$, $\mathcal{S}_j$) into account the uncertainties introduced by the bounded perturbation in (3.2).

Using Proposition 2.1 in Section 2.4.1, Chapter 2, we obtain the equivalent formulation for (3.15):

$$(x_i - x_j) \notin (\{-\mathcal{S}_i\} \oplus \mathcal{S}_j), \qquad i, j \in \mathcal{I},\ i \neq j \tag{3.16a}$$
$$x_i \notin (\{-\mathcal{S}_i\} \oplus \mathcal{O}_o), \qquad i \in \mathcal{I},\ o \in \mathcal{I}_o, \tag{3.16b}$$

Remark 3.3. The set relationships inclusions (3.16) assume that the obstacles are static, an alternative solution using parametrized polyhedra (see, for instance, [Loechner and Wilde, 1997] for detailed notions on the parametrized polyhedra) to describe the safety regions of the agents is presented in [Prodan et al., 2011b]. For guaranteeing that two (or more) agents do not superpose, the parametrized intersections of the invariant sets are considered and then, the domain for which the intersections are void is described [Olaru and Dumur, 2004]. However, we note that this approach is computationally demanding by the need of an explicit parameterized vertices construction. □

The main technical difficulty encountered in the formulation of the collision avoidance conditions (3.16) is the fact that, often, the feasible regions are non-convex and consequently, they cannot lead to an explicit expression of the feasible trajectories in the state-space. It is clear that this problem rises naturally from the separation conditions (3.16). The solution is to use the mixed-integer programming techniques (detailed in

[3]Whenever the time instant is clear we abuse the notation and denote the current state, $x_{d,i}(k)$, as $x_{d,i}$. The same notation is applied to the nominal system state and input vectors (3.3).

Section 2.5, Chapter 2), which allows us to express the original non-convex feasible region as a convex set in an extended space. A method for reducing the computational time was detailed in the previous chapter, where we propose a technique for making the time of computation P-hard in the number of Linear/Quadratic Programming (LP/QP) subproblems that have to be solved. In the following we present an illustrative example which considers a non-convex non-connected feasible region described through MIP. Elements like hyperplane arrangements and cell merging from the previous chapter are recalled and used here. We also detail the notable improvements in representation and reduction of computation time. These techniques will be implemented for dealing with the collision avoidance problem in the multi-agent setting.

Exemplification of a mixed-integer representation of the feasible region

For illustration purpose we consider the position component of an agent state to be constrained by 4 obstacles as shown in Figure 3.2, which may also represent safety regions around agents. We can also characterize these convex sets as interdicted regions in the context of multi-agent systems. The goal is to represent (in a tractable and minimized form) the non-convex feasible region using MIP as detailed in Section 2.5, Chapter 2.

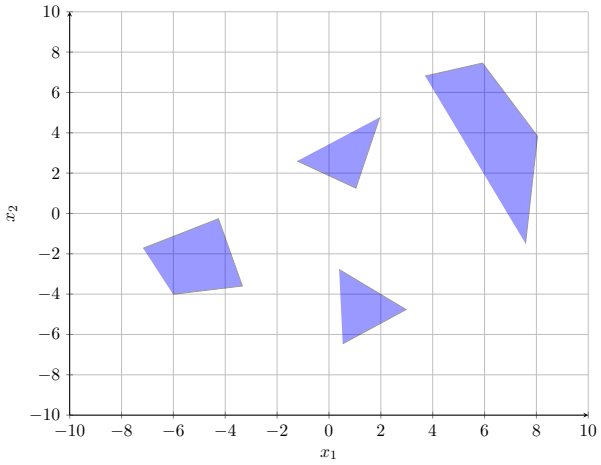
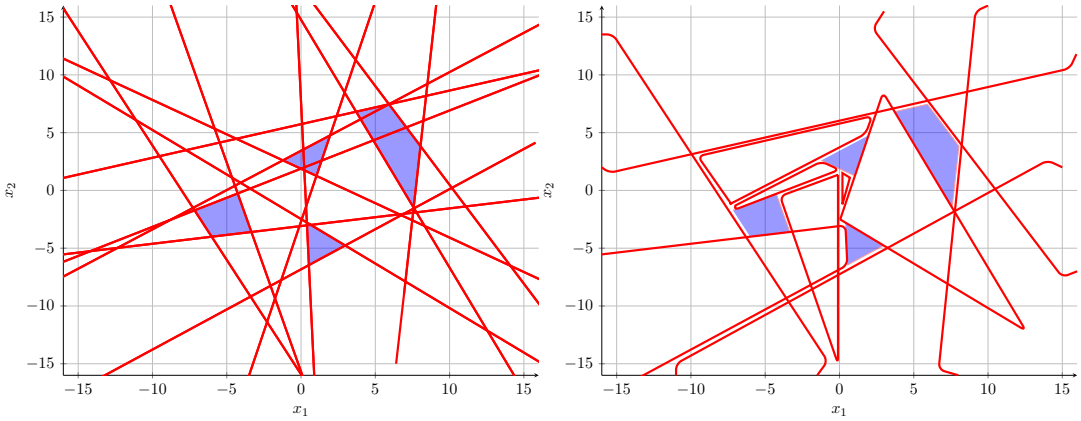


FIGURE 3.2: Prohibited regions in the state space.

Considering the 14 hyperplanes which describe the polyhedra associated with the obstacles, we have $\gamma(14) = 106$ regions obtained as in (2.29). Additionally, we observe that 10 of the cells will describe the interdicted regions, and the rest, $\gamma^b(14) = 96$ will describe the feasible region, as shown in (2.57). Furthermore, we apply the notions from Subsection 2.5.4, Chapter 2 to obtain a reduced representation for the feasible region as in (2.64). We observe that the number of cells is substantially reduced, from $\gamma^b(14) = 96$

to $\gamma^c(14) = 11$, which warrants in turn a reduction of the auxiliary binary variables from 7 to 4 that, for a worst case scenario, equals to an eightfold speed up. In Figure 3.3 (a), we depict the cells of (2.57) and the obstacles, while in Figure 3.3 (b) we show the covering (2.64) of merged cells.



(A) Partitioning with disjoint cells obtained as in (2.57)

(B) Partitioning with merged cells (2.64)

FIGURE 3.3: Cell partitioning of the feasible region.

A last aspect of interest is the *scalability* of the problem for a larger number of agents and/or obstacles. Obviously, the results depend greatly on the number of sets, the number of hyperplanes defining them and their relative positions. We depict in Table 3.1 a few cases (showing the number of obstacles, hyperplanes, and regions (2.40)–(2.42) and (2.53) respectively) and observe that the improvements are (at least for the examples considered) significant. We recall that in Table 3.1, $\#\mathbb{P}$ represents the number of obstacles, $\#\mathcal{H}_i$ denotes the number of hyperplanes, $\#\mathcal{A}_i$ denote all the cells, $\#\mathcal{B}_i$ all the feasible cells and $\#\mathcal{C}_i$ represent the merged cells.

$\#\mathbb{P}$	$\#\mathcal{H}_i$	$\#\mathcal{A}_i$	$\#\mathcal{B}_i$	$\#\mathcal{C}_i$
5	19	210	191	13
7	24	295	266	16
9	30	496	449	23

TABLE 3.1: Complexity analysis with increased numbers of obstacles.

Taking into account all the above considerations, we propose in the following to handle the multi-agent formation control problem.

3.3 A tight configuration of multi-agent formation

Once the description and representation of the obstacles and their geometry is realized, we are able to couple these properties with the dynamics and ask in a first instance a "static" question: *which is the desired formation?*, or in a more general sense, *which is the equilibrium point for the multi-agent system taking into account collision avoidance constraints?*.

3.3.1 Minimal configuration of the multi-agent formation

Suppose that the individual objective of each agent is the convergence to the origin of the state space. The global goal of clustering the agents as close as possible to the origin is realized through a minimal configuration for the group of agents (3.3) due to the adversary constraints of collision avoidance. We formulate the problem as an optimization-based problem, with a cost function defined by the sum of the square distances of each agent from the origin and the collision avoidance constraints (3.15):

$$\min_{x_i,\ i\in\mathcal{I}} \ \sum_{i=1}^{N_a} \|x_i\|_2^2, \tag{3.17}$$

$$\text{subject to: } (x_i - x_j) \notin (\{-S_i\} \oplus S_j), \quad \forall i,j \in \mathcal{I},\ i \neq j, \tag{3.18}$$

with $\mathcal{I}$ defined as in (3.1).

Note that the objective of points placement as close as possible to the origin is purely a geometrical condition and it can be easily described in a quantitative cost by the minimization of the sum of Euclidean norms as in (3.17). However, this constraint leads to some interesting implications from the point of view of the dynamics of the agents. This is because the final destination points are actually equilibrium points for each individual dynamics and may not be fixed points for their associated linear mapping (3.3). If the result of such an optimization problem is provided as reference for a multi-agent formation, the agents may not reach the target points or even if they passes through them, they will not converge to these a priori defined target positions. This will result in a chaotic behavior for the agents, with the actual final positions being indeterminate or inexistent (the case of limit cycles).

The solution needs to differentiate a "logistic solution" of the type (3.17)–(3.18) from a control solution which imposes an additional constraint upon the target points, namely,

to add fixed points constraints for their associated dynamics[4]. Therefore, the mixed-integer optimization problem (3.17) is expanded to:

$$\min_{(x_i,u_i),\ i\in\mathcal{I}} \sum_{i=1}^{N_a} \|x_i\|_2^2, \tag{3.19}$$

$$\text{subject to: } \begin{cases} (x_i - x_j) \notin (\{-S_i\} \oplus S_j), & \forall i,j \in \mathcal{I},\ i \neq j, \\ x_i = A_i x_i + B_i u_i, & \forall i \in \mathcal{I}. \end{cases} \tag{3.20}$$

Solving the non-convex optimization problem (3.19), a set of target positions and the associated control laws are obtained:

$$T = \left\{ (x_{f,1}, u_{f,1}), (x_{f,2}, u_{f,2}), \ldots, (x_{f,N_a}, u_{f,N_a}) \right\}, \tag{3.21}$$

where every pair $(x_{f,i}, u_{f,1})$ is a fixed state/input of the i^{th} agent. As seen in Chapter 2, a natural way to solve non-convex optimization problem is to rewrite it in a mixed-integer formulation. In such a case, the binary part resulting from the resolution of (3.19)–(3.20) provides a description of the active constraints.

Remark 3.4. Note that the optimization problem (3.19) will provide only pairs $(x_{f,i}, u_{f,i})$, $i \in \mathcal{I}$ which are also a fixed point for the considered dynamics, (3.3). Geometrically, this means that the points $x_{f,i}$ will find themselves on the associated subspaces spanned by $(I_n - A_i)^{-1}B_i$. In particular, if the agents have the same dynamics (i.e., homogeneous agents), they will have a common subspace over which to select the fixed points $x_{f,i}$. □

Exemplification for minimal configuration

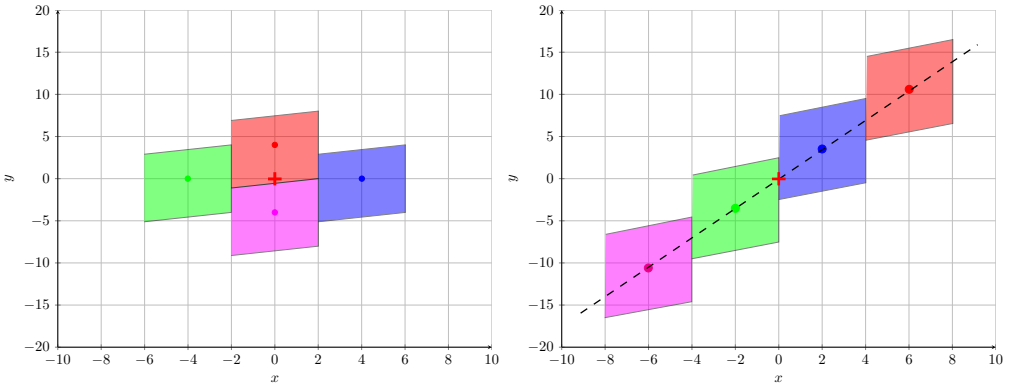
In order to illustrate the above multi-agent minimal configuration construction let us consider first $N_a = 4$ homogeneous agents described by the following dynamics in a two dimensional state-space:

$$A_i = \begin{bmatrix} 0.97 & -0.04 \\ 0.81 & 0.42 \end{bmatrix}, \quad B_i = \begin{bmatrix} 0.30 \\ 0.72 \end{bmatrix},$$

with $i = 1, \ldots, 4$. Similarly with the example illustrated in Figure 3.1 and as explained in Section 3.1, we construct safety regions around the agents (which in this particular case are identical) in order to proceed with the resolution of the minimal configuration problem. By solving the optimization problem (3.17)–(3.18) we obtain the set of target positions $T = \{(-4, 0), (0, -4), (0, 4), (4, 0)\}$ and the minimal configuration as described in Figure 3.4 (a).

[4]Once this additional condition (linked to the dynamics) is fulfilled we can guarantee that a trajectory can actually be steered towards that point and once there, it can remain in it.

Furthermore, by solving the optimization problem (3.19)–(3.20) we obtain the set of target positions $T = \{(6, 10.7, 2, 03), (2, 3.91, 0.7), (-2, -3.91, -0.7), (-6, -10.7, -2, 03)\}$ and the minimal configuration as depicted in Figure 3.4 (b). Usually, the collection of feasible fixed points is degenerate with respect to the state-space (since it depends on the input which is usually lower dimensional than the state). Consequently, the fixed points can stay only on a certain subspace of the state-space (as detailed also in Remark 3.4). To exemplify, consider Figure 3.4 (b) where, for the case of agents with the same dynamics the equilibrium points of (3.21) stay on the same subspace, which, in this particular case is a line passing through the origin (i.e., $x_{f,i} = [2.91 \quad 5.35]^T u_{f,i}$, with $i = 1, \ldots, 4$). This means that the result of optimization (3.19) gives a "beans on a string" formation[5].



(A) Minimal configuration of 4 homogeneous agents when solving (3.17)–(3.18).

(B) Minimal configuration of 4 homogeneous agents when solving (3.19)–(3.20).

FIGURE 3.4: Minimal configuration of 4 homogeneous agents with their safety regions.

Following the same computations, in Figure 3.5 we consider the minimal configuration of $N_a = 4$ heterogeneous agents in a two dimensional state-space described by:

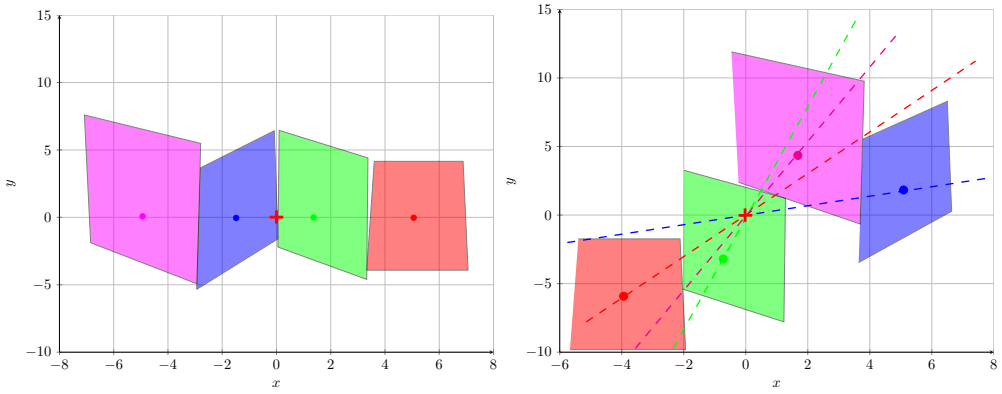
$$A_1 = \begin{bmatrix} 1.15 & -0.14 \\ 0.81 & 0.44 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0.71 \\ 0.22 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 5.54 & -12.65 \\ 1.77 & 3.94 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0.04 \\ 0.75 \end{bmatrix},$$
$$A_3 = \begin{bmatrix} 0.68 & 0.05 \\ -0.07 & 0.91 \end{bmatrix}, \quad B_3 = \begin{bmatrix} 0.13 \\ 0.94 \end{bmatrix}, \quad A_4 = \begin{bmatrix} 0.45 & 0.20 \\ -0.54 & 1.14 \end{bmatrix}, \quad B_4 = \begin{bmatrix} 0.08 \\ 0.85 \end{bmatrix}.$$

Safety regions around each of the agents are constructed as detailed in Section 3.1. Then, by solving the optimization problem (3.17)–(3.18) we obtain the set of target positions

[5]Note that we consider here a two dimensional system with a scalar input because it provides illustrative results, i.e., the subspace defining the fixed points is a line and thus, easily representable. More precisely the dashed line denotes the subspace of fixed points (note that it passes through the origin, regardless of the dynamics, the origin is always an equilibrium point in the LTI case.

$T = \{(-4.48, 0), (-1.8, 0), (1.78, 0), (4.5, 0)\}$ and the minimal configuration illustrated in Figure 3.5 (a).

Furthermore, by solving the optimization problem (3.19)–(3.20) we obtain the set of target positions $T = \{(-3.94, -5.9, -0.32), (5.09, 1.83, -0.01), (-0.72, -3.19, -0.33), (1.68, 4.34, 0.3)\}$ and the minimal configuration as depicted in Figure 3.5 (b). The placement of the target positions (i.e., the equilibrium points) is no longer evident, but still, each of the agents moves only along its equilibrium points line which passes through the origin, see Figure 3.5 (b).



(A) Minimal configuration of 4 heterogeneous agents when solving (3.17)–(3.18).

(B) Minimal configuration of 4 heterogeneous agents when solving (3.19)–(3.20).

FIGURE 3.5: Minimal configuration of 4 heterogeneous agents with their safety regions.

An important observation for the case of homogeneous agents is that an interchange in the assignment of a target position in between the agents will not modify the global optimality (because the safety regions are identical) with respect to the optimization problem (3.19)–(3.20). This means that the optimum is not unique, although all the optimal solutions are equivalent from the dynamical point of view. This allows us to detail in the following the procedure of optimal task assignment. Introducing this procedure during the real-time functioning of the agents will increase the flexibility of the scheme and reduce the overall optimization cost.

3.3.2 Task assignment formulation

In the particular case of homogeneous[6] agents (understood here as agents with the same safety regions) we can intercalate an additional step in the control mechanism.

[6]In the heterogeneous case the reassignment of the final destination points is no longer feasible since the swapping of the safety regions will result in collisions of the agents.

Since the agents have the same safety regions, we can adapt in real-time the response to the question, "who goes where in the minimal configuration?", as computed in the previous subsection. This is equivalent with finding the best permutation over the set of the final positions in the target formation, T from (3.21). This idea is not new in the logistics problems and, is known under the name of optimal assignment problem encountered in the field of combinatorial optimization [Hallefjord Kurt, 1993], [Osiadacz, 1990]. However, in the multi-agent control framework, the optimal assignment in real-time brings a novel (dynamical) dimension in terms of time-varying target.

If one associates a cost with the assignment of agent j to target $x_{f,i}$ as c_{ij}, the problem of finding the best assignment is defined as:

$$\min_{\delta_{ij},\ i,j\in\mathcal{I}} \sum_{i=1}^{N_a}\sum_{j=1}^{N_a} c_{ij}\delta_{ij}, \tag{3.22}$$

$$\text{subject to: } \begin{cases} \sum\limits_{i=1}^{N_a} \delta_{ij} = 1, & \forall i,j \in \mathcal{I}, \\ \sum\limits_{j=1}^{N_a} \delta_{ij} = 1, & \forall i,j \in \mathcal{I}, \\ \delta_{ij} \in \{0,1\}, & \forall i,j \in \mathcal{I}, \end{cases} \tag{3.23}$$

where δ_{ij} are the decision variables:

$$\delta_{ij} = \begin{cases} 1\ , & \text{if target } x_{f,i} \text{ is assigned to agent } j, \\ 0\ , & \text{otherwise,} \end{cases} \tag{3.24}$$

and $\mathcal{I}$ defined as the collection of all agents indices (3.1). These binary variables ensure that each agent is assigned to one unique target position.

The problem is defined by the choice of the cost weights c_{ij}, the simplest way is to choose it as the distance between the actual position of agent j and the desired target position in the formation. Hence, the problem would be to determine the minimal distance that an agent has to travel to establish the optimal assignment in the specified formation. A more insightful way is to use the unconstrained dynamics (3.3) of the agents to describe the cost of reaching from the initial position to the desired position. Then, c_{ij} can be described by a weighted norm:

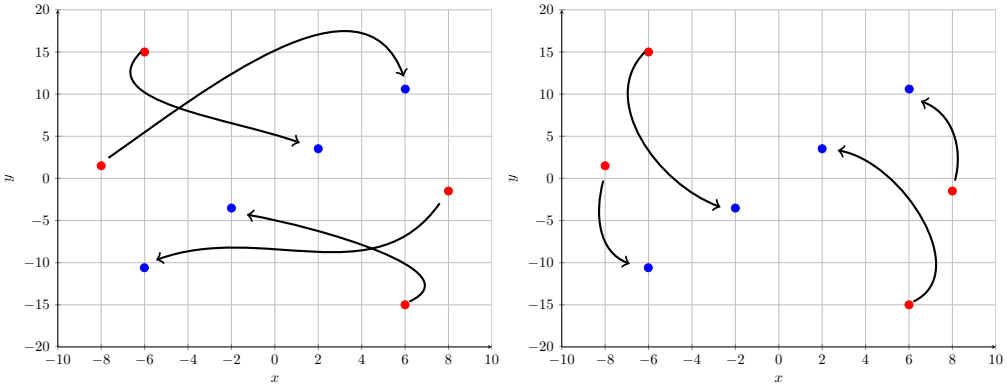
$$c_{ij} = (x_j - x_{f,i})^T P (x_j - x_{f,i}), \quad \forall i,j \in \mathcal{I} \tag{3.25}$$

with the matrix $P = P^T \geq 0$ given by the Lyapunov function or the infinite time cost-to-go, as long as the agents follow the unconstrained optimum through the control action:

$$u_j = -K_j(x_j - x_{f,i}) + \bar{u}_j, \quad \forall i,j \in \mathcal{I},\ i \neq j, \tag{3.26}$$

where $\bar{u}_j$ is chosen such that $x_{f,j} = A_j x_{f,i} + B_j \bar{u}_j$, with $\bar{u}_j = B_j^{-1}(I - A_j)x_{f,i}$, if the matrix B_j is invertible (or the alternative pseudo-inverse which allows the definition of a fixed point for the nominal trajectory). This optimization problem can be reduced to a simple LP problem, hence it can be efficiently computed.

Note that, the task assignment can be seen as either an off-line or on-line procedure. Off-line, it simply means that we assign the agents to the target points at the beginning and then, they remain in the same configuration. On the other hand, the on-line approach means that at each time instant (or several time instances) the assignment is re-evaluated according to the current position.



(A) Assignment with a non-minimal cost.

(B) Assignment with a minimal cost.

FIGURE 3.6: Exemplification for optimal task assignment.

Exemplification for optimal task assignment

Consider the set of 4 target positions (blue dots) obtained as in Figure 3.4 (b) (i.e., $T = \{(6, 10.7), (2, 3.91), (-2, -3.91), (-6, -10.7)\}$) and a set of 4 initial positions (red dots) for the agents, depicted in Figure 3.6 (a) with a random assignment. Figure 3.6 (b) illustrates the optimal assignment for comparison. It can be observed that the cost of reaching the target positions is smaller in the second case.

Finally, before passing to the on-line optimization-based control problem for the feedback control design, we present in Figure 3.7 a schematic view of the methods presented above. The interdependence between the methodologies and mathematical/optimization formulations is stressed with a particular decomposition in on-line vs. off-line components.

In the following we detail the on-line part of the control problem. The centralized MPC optimization problem is described first and subsequently used in a latter section in order

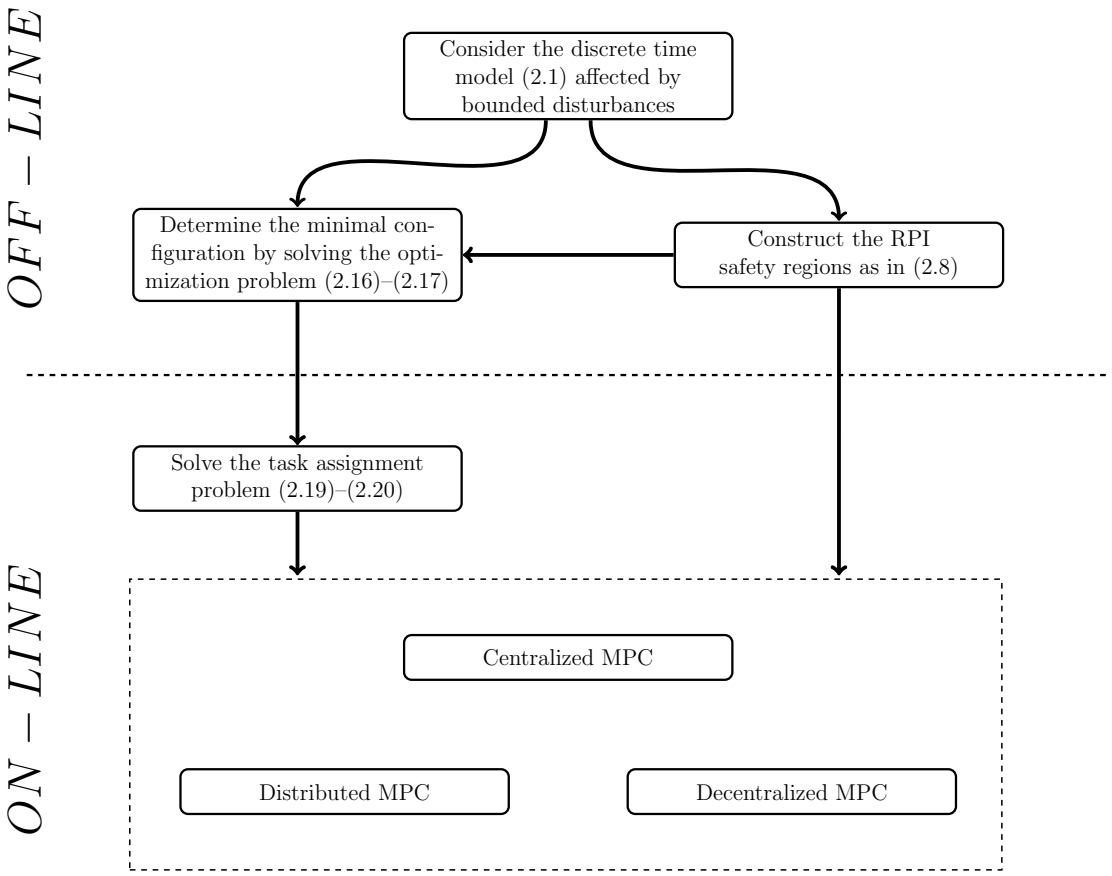


FIGURE 3.7: General multi-agent formation control approach

to construct and analyze a particular distributed MPC approach, all by exploiting the MIP formalism provided in the previous chapter.

In the last part of the present chapter we propose a control strategy which is a combination of MPC and Potential Field approach. The solution is decentralized as the steering policy for each agent is based only on local state information from its nearest neighbors.

3.4 Centralized MPC

In the centralized approach perfect knowledge of each agent dynamics is available to all the other agents. The agents are seen as a single block where there exists perfect communication and the control action is designed by taking into account the states and constraints at a global level. The main advantage of the centralized approach is that it

provides the best performances with the price of an excessive communication between the agents.

As it can be seen from Figure 3.8 there exists a centralized control mechanism which takes as inputs information (past inputs and states) from all agents and them provides each of them with updated inputs.

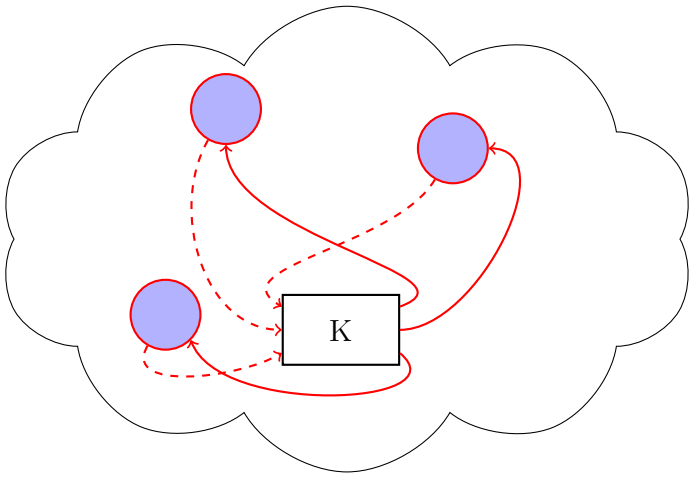


FIGURE 3.8: Centralized MPC architecture.

3.4.1 Centralized prediction model

Let us consider the set of N_a constrained sub-systems (agents) as a *global system* defined as:

$$\mathbf{x}_{\mathcal{I}}(k+1) = \mathbf{A}_{\mathcal{I}}\mathbf{x}_{\mathcal{I}}(k) + \mathbf{B}_{\mathcal{I}}\mathbf{u}_{\mathcal{I}}(k), \tag{3.27}$$

with the corresponding vectors which collects the states and the inputs of each individual nominal system (3.3) at time k:

$$\mathbf{x}_{\mathcal{I}}(k) = [x(k)_1^T|\cdots|x(k)_{N_a}^T]^T, \quad \mathbf{u}_{\mathcal{I}}(k) = [u(k)_1^T|\cdots|u(k)_{N_a}^T]^T \tag{3.28}$$

and the matrices $\mathbf{A}_{\mathcal{I}} \in \mathbb{R}^{N_a n \times N_a n}$, $\mathbf{B}_{\mathcal{I}} \in \mathbb{R}^{N_a n \times N_a m}$ which describe the model:

$$\mathbf{A}_{\mathcal{I}} = diag[A_1, \ldots, A_{N_a}], \quad \mathbf{B}_{\mathcal{I}} = diag[B_1, \ldots, B_{N_a}]. \tag{3.29}$$

and the set $\mathcal{I}$ defines collection of all agents indices as in (3.1).

So far the subsystems belonging to the global system are completely decoupled from the dynamical point of view. Also, perfect knowledge of each agent dynamics described by equation (3.3) is available to all the other agents. Furthermore, the global model will be used in a predictive control context which permits the use of non-convex constraints

for collision avoidance behavior, this being the principal coupling effect. In order to describe a minimal configuration for the group of agents, the result of a mixed-integer optimization problem is supposed to be available. The results will be further used such that the agents will converge towards the predefined formation without colliding.

3.4.2 Centralized optimization-based control problem

The goal is to drive the agents to a minimal configuration (3.21) (if possible, adapting it on-line via the optimization (3.22)) while in the same time avoiding collisions along their evolution towards the formation. To this end, we consider an optimal control problem for the global system defined in (3.27) where the cost function and the constraints couple the dynamic behavior of the individual agents. Also, perfect knowledge of each agent dynamics described by equation (3.3) is available to all the other agents. Consequently, the global model will be used in a predictive control context which permits the use of non-convex constraints for collision avoidance behavior.

A finite receding horizon implementation is typically based on the solution of an open-loop optimization problem. We consider a cost function $V_n(\mathbf{x},\mathbf{u}) : \mathbb{R}^{N_a\cdot n}\times\mathbb{R}^{N_a\cdot m}\to\mathbb{R}$ which aims at maintaining a formation, following a reference path or simply to gather the agents towards the minimal formation. The centralized optimization problem under collision avoidance constraints is formulated as follows[7]:

$$\mathbf{u}_{\mathcal{I}}^{*} = \arg \min_{\mathbf{u}_{\mathcal{I}}(k|k),\ldots,\mathbf{u}_{\mathcal{I}}(k+N_p-1)|k)} V_n(\mathbf{x}_{\mathcal{I}}(k|k),\mathbf{u}_{\mathcal{I}}(k|k),\ldots,\mathbf{u}_{\mathcal{I}}(k+N_p-1|k) \tag{3.30a}$$

$$\text{subject to:} \quad \begin{cases} \mathbf{x}_{\mathcal{I}}(k+s+1|k) = \mathbf{A}_{\mathcal{I}}\mathbf{x}_{\mathcal{I}}(k+s|k) + \mathbf{B}_{\mathcal{I}}\mathbf{u}_{\mathcal{I}}(k+s|k), \quad s=0,\ldots,N_p-1, \\ (P_i-P_j)\,\mathbf{x}_{\mathcal{I}}(k+s|k) \notin (\{-\mathcal{S}_i\}\oplus\mathcal{S}_j)\,,\ \forall i,j\in\mathcal{I},\ i\neq j,\ s=1,\ldots,N_p, \end{cases} \tag{3.30b}$$

where matrix $P_i \triangleq [0\ \ldots\ \underbrace{I}_{i}\ \ldots\ 0]$ projects the individual agent state x_i from the extended state $\mathbf{x}$ (i.e., $x_i(k) = P_i\mathbf{x}(k)$, $i\in\mathcal{I}$). With an abuse of notation we use the non-convex constraints in (3.30b) rewritten with the help of MIP techniques (as in Section 2.5, Chapter 2) into the following form:

$$\mathbf{x}_{\mathcal{I}}(k+s|k)\notin\mathbb{S}, \quad \mathcal{C}(\mathbb{S}) = \bigcup_{l=1,\ldots,\gamma^c(N)} C_l,\ s=1,\ldots,N_p, \tag{3.31}$$

[7]Using the elements provided in Section 2.5, Chapter 2, the computational complexity of (3.30) can be assessed to a polynomial number of QP problems.

where $\mathbb{S}$ represents the union of convex (bounded polyhedral) safety regions $\mathbb{S} \triangleq \bigcup_{i\in\mathcal{I}} \mathcal{S}_i$, with $\mathcal{S}_i$ the RPI sets with respect to the dynamics in presence of additive disturbance defined as in (3.9).

The optimization problem (3.30) requires the minimization of the cost function over a finite prediction horizon N_p. From the optimal sequence of inputs $\mathbf{u}_\mathcal{I}(k)^*, \ldots, \mathbf{u}_\mathcal{I}(k+N_p-1|k)^*$ the first control input, $\mathbf{u}(k)^*$, is selected and applied to the centralized system, thus closing the loop. In order to assure that the target positions (3.21) are reached, we require a cost function which is minimized in the destination points (and not the origin):

$$\begin{aligned} V_n(\mathbf{x}_\mathcal{I}(k|k), \mathbf{u}_\mathcal{I}(k|k), \ldots, \mathbf{u}_\mathcal{I}(k+N_p-1|k) = \\ (\mathbf{x}_\mathcal{I}(k+N_p|k) - x_{f,i(k)})^T \mathbf{P} (\mathbf{x}_\mathcal{I}(k+N_p|k) - x_{f,i(k)}) + \\ + \sum_{s=1}^{N_p-1} (\mathbf{x}_\mathcal{I}(k+s|k) - x_{f,i(k)})^T \mathbf{Q} (\mathbf{x}_\mathcal{I}(k+s|k) - x_{f,i(k)}) + \\ + \sum_{s=0}^{N_p-1} (\mathbf{u}_\mathcal{I}(k+s|k)^T - u_{f,i(k)}) \mathbf{R} (\mathbf{u}_\mathcal{I}(k+s|k) - u_{f,i(k)}), \end{aligned} \tag{3.32}$$

where $(x_{f,i(k)}, u_{f,i(k)})$ represents the optimal target positions and the associated control laws at current time k, $i \in \mathcal{I}$. Here $\mathbf{Q} = \mathbf{Q}^T \succeq 0$, $\mathbf{R} \succ 0$ are the weighting matrices with appropriate dimensions, $\mathbf{P} = \mathbf{P}^T \succeq 0$ defines the terminal cost and N_p denotes the length of the prediction horizon. Through the task assignment mechanism, the association between an agent and a target position may change at any moment in time updated accordingly. This means that the affine terms defining the cost function must be updated accordingly.

Remark 3.5. The conditions for the centralized obstacle avoidance problem can be rewritten in a similar way as in (3.30b):

$$P_i \mathbf{x} \notin (\{-\mathcal{S}_i\} \oplus \mathcal{O}_o), \quad i \in \mathcal{I},\ o \in \mathcal{I}_o, \tag{3.33}$$

with the matrix P_i defined as in (3.30b) and $\mathcal{I}$, $\mathcal{I}_l$ defined as in (3.1) and (3.13), respectively. □

Let us summarize in the following algorithm the receding horizon strategy together with task assignment mechanism:

Due to the fact that we use invariant sets, steps 1 and 2 can be executed in an off-line procedure. In the on-line part of the algorithm, we apply a finite horizon trajectory optimization: in step 5 we execute a task assignment if possible (only if the safety regions are identical) and then proceed with the actual computation of the receding horizon control (step 7). Finally, the first component of the resulting control sequence is effectively applied to the global system (step 8) and the optimization procedure is

Algorithm 3.1: Centralized scheme strategy for the tight formation control problem

Input: initial positions of the agents

-construct the safety regions associated to each agent;
-construct the minimal configuration as in (3.19)–(3.20);
for $k = 1 : k_{max}$ **do**
  if *the safety regions are identical* **then**
    -execute task assignment $x_i(k) \to x_{f,i}$ as in (3.22);
  end
  -find the optimal control action $u_{\mathcal{I}}^*$ as in (3.30);
  -compute the next value of the state:

$$\mathbf{x}_{\mathcal{I}}(k+1) = \mathbf{A}_{\mathcal{I}}\mathbf{x}_{\mathcal{I}}(k) + \mathbf{B}_{\mathcal{I}}\mathbf{u}_{\mathcal{I}}(k).$$

end

reiterated using the available measurements based on the receding horizon principle [Mayne et al., 2000].

Remark 3.6. Although functional, this scheme will not scale favorably with an increased number of agents or a large prediction horizon due to the numerical difficulties of the optimization in large dimensional spaces. In particular, the mixed-integer algorithms are very sensitive to the number of binary auxiliary variables. In this case a distributed approach is to be envisaged in order to minimize the numerical computations. □

Remark 3.7. Note that although desirable, an increase in the length of the prediction horizon is not always practical, especially when using mixed-integer programming. We observed that a two-stage MPC, where in the first stage a task assignment procedure is carried and in the second, the usual optimization problem is solved offers good performances with a reduced computational effort. □

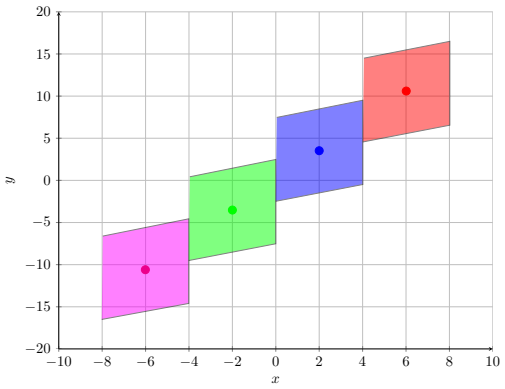
3.4.3 Exemplification for the convergence towards the tight formation

We apply the receding horizon scheme (3.30) for the global system with a prediction horizon $N_p = 2$ and the tunning parameters $\mathbf{P} = 5I_4$, $\mathbf{Q} = I_4$, $\mathbf{R} = I_2$. The optimal trajectories for the agents are obtained such that the set of target points is reached through task optimization and under (anti-collision) state constraints. The effectiveness of Algorithm 3.1 is confirmed by the simulation depicted in Figure 3.9 (b), where the evolution of the agents is represented at three different time instances. The agents successfully reach their target positions in the predefined formation (illustrated again in Figure 3.9 (a)) without violating the constraints and with a minimum cost.

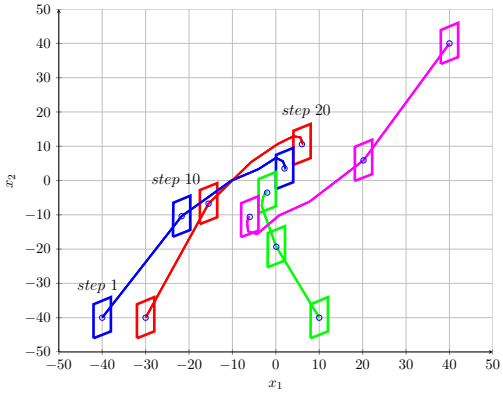
For comparison purposes we execute Algorithm 3.1 with and without the task assignment stage. As it can be seen in Figure 3.9 (c) for a prediction horizon of $N_p = 2$ and without

the task assignment procedure, the agents do not converge to the desired configuration (two agents, depicted in blue and red, respectively switch places with respect to the predefined optimum, which means that the cost function only decrease to a constant non-zero value). The poor performances of example Figure 3.9 (c) are due to the "untuned" prediction horizon. Choosing a small prediction horizon makes the control "blind" in the sense that the trajectories "do not see" the need for a a control effort in order reach the desired configuration. Of course, these problems can be mitigated with an increase of the prediction horizon. Note that if the prediction horizon is long enough (in this particular case $N_p = 8$) the desired configuration is achieved but the computational complexity of the mixed-integer optimization problem (3.30) increases significantly.

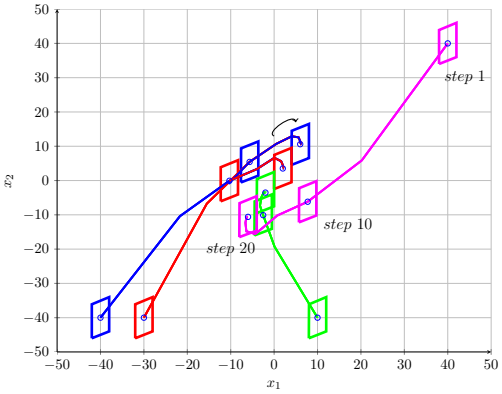
For a second illustration, let us consider $N_a = 4$ heterogeneous agents with different values of the parameters m_i and μ_i, $i = 1, \ldots, 4$. The prediction horizon is $N_p = 7$ and



(A) Target configuration of 4 homogeneous agents.

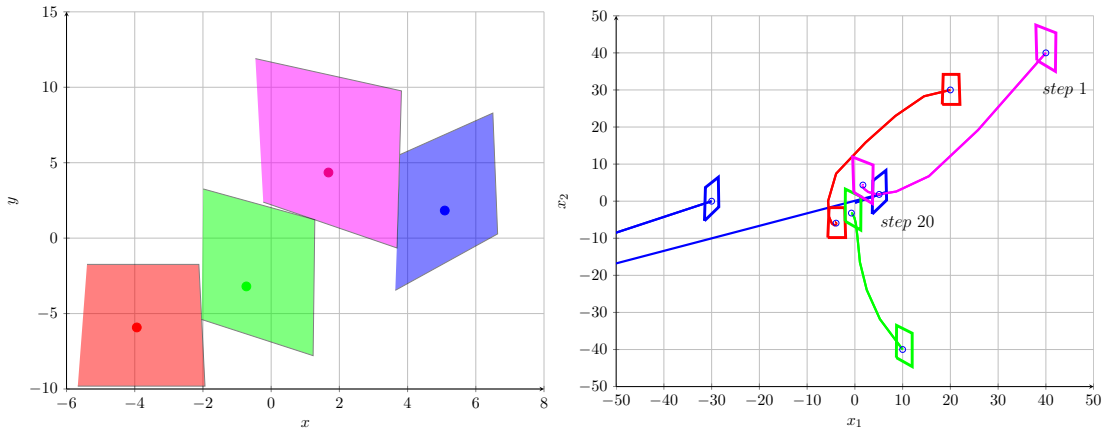


(B) Actual motion of 4 homogeneous agents with task assignment.



(C) Actual motion of 4 homogeneous agents without task assignment.

FIGURE 3.9: Tight formation of 4 homogeneous agents.



(A) Target configuration of 4 heterogeneous agents.

(B) Actual motion of 4 heterogeneous agents.

FIGURE 3.10: Tight formation of 4 heterogeneous agents.

the tunning parameters are $\mathbf{P} = 50I_4$, $\mathbf{Q} = I_4$, $\mathbf{R} = I_2$. Following the same procedure we depict in Figure 3.10 (a) the agents with the associated safety regions in a minimal configuration. Furthermore, in Figure 3.10 (b) we illustrate the evolution of the agents at two different time instances. Note that, in this particular case, the task assignment problem can not be employed since the safety regions of the agents are different and moreover, the target positions will be changing at each time instant.

Note that the problem at hand might still be difficult to solve by meeting the real-time constraints, even more so when a MPC scheme is applied (both the dimension of the solution space and the difficulty of describing the feasible region become larger with an increase in the length of the prediction horizon). Naturally, distributed MPC can be used to tackle this problem and the forthcoming section aims to offer a MIP solution in the context of distributed predictive control.

3.5 Distributed MPC

Recalling equation (2.56) from Section 2.5.2, Chapter 2 we note that, the increase in space dimension and number of obstacles reflects negatively into the computational complexity of the MIP problem. More precisely, the increase in the number of cells (as in (3.31)) makes their enumeration more difficult and further increases (even after applying merging algorithms) the number of necessary binary variables. Therein lies the main issue which plagues the centralized formulation: the mixed-integer algorithms increase in worst case situations exponentially with respect to the number of binary auxiliary variables and the computation reliability decreases for high dimensions (we

emphasize here the negative effects of dimension increases for MIP techniques, but the same holds for any optimization problem which deals with centralized implementation, whenever the number of agents increases).

We shall concentrate in this section on applying distributed optimization concepts to the formation multi-agent problem with collision avoidance capacities. To alleviate the issues plaguing the centralized description we propose two enhancements:

- to reduce the complexity of the problem and the dimension of the space in which the problem is solved, we consider a distributed MPC approach. We partition the agent collection into neighborhoods and compute the solutions locally while using information provided by the other neighborhoods. The stability of the overall formation and a reasonable performance of the agents also needs to be verified;
- to reduce the complexity of the feasible region representation we decompose only the "visible" part of the feasible region (i.e., the region which is reachable along the prediction horizon of the agents from a given neighborhood).

Figure 3.11 illustrates the distributed approach that we adopt here, that is, the agents are partitioned into neighborhoods. At the level of such a neighborhood there exists a control mechanism which provides inputs for the agents of the group. Moreover, each controller assumes a certain behavior of the other neighborhoods. In order to reach a consensus, they exchange communication and converge towards a common solution.

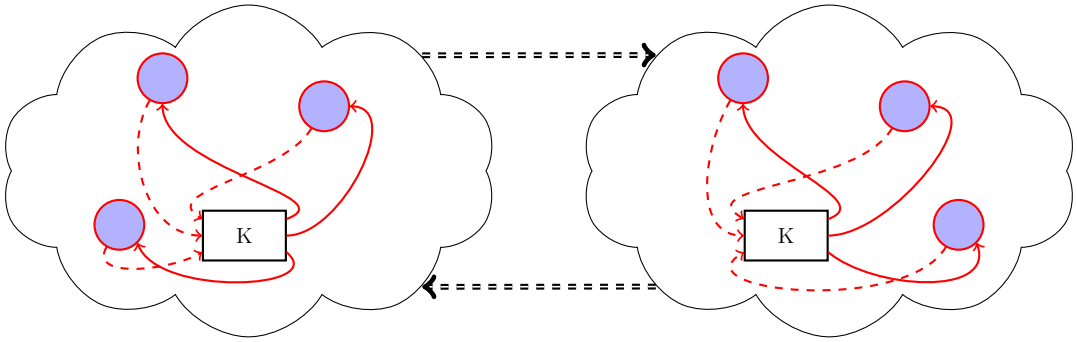


FIGURE 3.11: Distributed MPC architecture.

In the forthcoming subsections we will detail these improvements and show that the computational load is reduced significantly while the performance of the scheme remains within acceptable bounds.

3.5.1 Distributed system description

For further use, we introduce the following definition of a neighborhood of agents.

Definition 3.1. *Let* $\mathbb{N} \triangleq \bigcup_i \{\mathcal{N}_i\}$ *be the collection of neighborhoods* $\mathcal{N}_i \subseteq \mathcal{I}$ *which are considered to be* disjoint *(for any* $i \neq j$*,* $\mathcal{N}_i \cap \mathcal{N}_j = \emptyset$*) and to cover* $\mathcal{I}$ *(for any* $i \in \mathcal{I}$ *there exists* j *such that* $i \in \mathcal{N}_j$*), with* $\mathcal{I}$ *defined as in* (3.1). □

In the distributed case, the optimization problem is solved at the level of each neighborhood and then, the results are communicated externally to the other neighborhoods. Usually the procedure is repeated until a consensus is reached (sometimes the iterations can be avoided by, e.g., assuming a hierarchical structure). The dynamics corresponding to a neighborhood $\mathcal{N}_i$ are given by:

$$\mathbf{x}_{\mathcal{N}_i}(k) = \mathbf{A}_{\mathcal{N}_i}\mathbf{x}_{\mathcal{N}_i}(k) + \mathbf{B}_{\mathcal{N}_i}\mathbf{u}_{\mathcal{N}_i}(k), \tag{3.34}$$

with the corresponding vectors which collects the states and the inputs of each individual nominal system (3.3) at time k as follows:

$$\mathbf{x}_{\mathcal{N}_i}(k) = \left[\dots|x_j(k)^T|\dots\right]^T, \quad \mathbf{u}_{\mathcal{N}_i}(k) = \left[\dots|u_j(k)^T|\dots\right]^T, \quad j \in \mathcal{N}_i, \tag{3.35}$$

and the matrices $\mathbf{A}_{\mathcal{N}_i} \in \mathbb{R}^{\mathcal{N}_i n \times \mathcal{N}_i n}$, $B_{\mathcal{N}_i} \in \mathbb{R}^{\mathcal{N}_i n \times \mathcal{N}_i m}$ which describe the model:

$$\mathbf{A}_{\mathcal{N}_i} = diag[\dots, A_j, \dots], \quad B_{\mathcal{N}_i} = diag[\dots, B_j, \dots], \quad j \in \mathcal{N}_i, \tag{3.36}$$

for all $i \in \mathcal{I}$, with $\mathcal{I}$ defined as in (3.1).

For each optimization involving the agents of a given neighborhood, the feasible region can be obtained from conditions (3.16) by selecting only the constraints involving the agents of the current neighborhood and taking the other agents (of the remaining neighborhoods) as additional obstacles. This approach has the drawback of necessitating a continuous recalculation of decomposition (2.64) from Section 2.5.4, Chapter 2. An alternative solution which uses the previously calculated $\mathcal{C}(\mathbb{S})$ to compute the feasible region corresponding to $\mathcal{N}_i$ is proposed in the following statement.

Proposition 3.1. *Let the variable* $\mathbf{x}_{\mathcal{N}_i} \triangleq \begin{bmatrix} x_{i_1}^T & x_{i_2}^T & \dots \end{bmatrix}^T$ *denote the agents of the neighborhood* $\mathcal{N}_i$ *and* $\mathbf{x}_{\mathcal{I}\setminus\mathcal{N}_i}$ *denote the remaining agents. With the notation of* (3.30b)–(3.31) *and assuming that* $\mathbf{x}_{\mathcal{I}\setminus\mathcal{N}_i}$ *is bounded by a set* $\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}$ *we have that the feasible region characterizing* $\mathbf{x}_{\mathcal{N}_i}$ *is defined as:*

$$\mathcal{C}(\mathbb{S})|_{\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}} = \bigcup_{l=1,\dots,\gamma^c(N)} C_l|_{\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}}, \tag{3.37}$$

with

$$C_l|_{\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}} \triangleq \left\{ \mathbf{x}_{N_i} : \begin{bmatrix} \mathbf{x}_{\mathcal{N}_i} \\ \mathbf{x}_{\mathcal{I}\setminus\mathcal{N}_i} \end{bmatrix} \in C_l, \ \forall x_{\mathcal{I}\setminus\mathcal{N}_i} \in \mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i} \right\}. \tag{3.38}$$

□

Proof. The proof is constructive. By intersecting every cell C_l with $\mathbb{R}^{n\cdot|\mathcal{N}_i|} \times \mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}$ and then projecting along the $\mathbf{x}_{\mathcal{N}_i}$ subspace we obtain the restrictions $C_l|_{\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}}$, thus concluding the proof. □

Recalling the results from Section 2.5, Chapter 2 and the illustrative example provided in Section 3.2, we observe that in the centralized description, the feasible space for the extended state (3.27) is obtained as the union of cells C_l in (3.31). For the distributed approach we have a reduced feasible space where the variables are the states of the agents of the current neighborhoods and the rest of the agents are considered as obstacles. Of course, we could use the techniques of Section 2.5 to obtain the feasible region but this would imply a continuous recalculation of the region as the "mobile obstacles" (i.e., the other agents) will move from one instant to the other. Instead, we use the centralized feasible region (3.31) and observe (as shown above) that the feasible region characterizing neighborhood $\mathcal{N}_i$ can be obtained through simple set operations (as seen in (3.37)–(3.38)). Alternatively, we can say that this approach is an "explicit" one since it requires the off-line computation of a region (3.31), which is then particularized for various values of the states of the agents which *are not* in the current neighborhood.

3.5.2 Distributed optimization-based control problem

As detailed in the introduction, the distributed approach has obvious computational benefits. Consequently, we reformulate problem (3.30) into a distributed form. For the agents of neighborhood $\mathcal{N}_i$, the local optimization problem becomes:

$$\mathbf{u}^*_{\mathcal{N}_i} = \arg \min_{\mathbf{u}_{\mathcal{N}_i}(k|k),\ldots,\mathbf{u}_{\mathcal{N}_i}(k+N_p-1)|k)} \sum_{s=0}^{N_p-1} V_{\mathcal{I}\setminus\mathcal{N}_i}(\mathbf{x}_{\mathcal{N}_i}(k+s|k), \mathbf{u}_{\mathcal{N}_i}(k+s|k)) \tag{3.39a}$$

$$\text{s.t.:} \quad \mathbf{x}_{\mathcal{N}_i}(k+s+1|k) = \mathbf{A}_{\mathcal{N}_i}\mathbf{x}_{\mathcal{N}_i}(k+s|k) + \mathbf{B}_{\mathcal{N}_i}\mathbf{u}_{\mathcal{N}_i}(k+s|k), \quad s = 0,\ldots,N_p-1, \tag{3.39b}$$

$$\mathbf{x}_{\mathcal{N}_i}(k+s|k) \in \mathcal{C}(\mathbb{S})|_{\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}}, \quad s = 1,\ldots,N_p. \tag{3.39c}$$

The use of indexing "$\mathcal{N}_i$" in (3.39) is to be understood as in Proposition 3.1, e.g., $\mathbf{A}_{\mathcal{N}_i}$ denotes the concatenation (block-diagonal in this case) of state matrices A_j where the indices j are found in the neighborhood $\mathcal{N}_i$.

The local cost function $V_{\mathcal{I}\setminus\mathcal{N}_i}(\mathbf{x}_{\mathcal{N}_i}, \mathbf{u}_{\mathcal{N}_i}) : \mathbb{R}^{|\mathcal{N}_i|\cdot n} \times \mathbb{R}^{|\mathcal{N}_i|\cdot m} \to \mathbb{R}$ is defined as

$$V_{\mathcal{I}\setminus\mathcal{N}_i}(\mathbf{x}_{\mathcal{N}_i}, \mathbf{u}_{\mathcal{N}_i}) = \min_{\substack{\mathbf{u}_{\mathcal{I}\setminus\mathcal{N}_i}\in\mathcal{U}_{\mathcal{I}\setminus\mathcal{N}_i} \\ \mathbf{x}_{\mathcal{I}\setminus\mathcal{N}_i}\in\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}}} V(\mathbf{x}, \mathbf{u}), \tag{3.40}$$

where $\mathcal{U}_{\mathcal{I}\setminus\mathcal{N}_i}$ denotes the values taken by the inputs $u_{\mathcal{I}\setminus\mathcal{N}_i}$ (similarly with the definition of set $\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}$).

Remark 3.8. The use of operator "min" assumes a cooperative approach (the agents exterior to the current neighborhood will try to accommodate the inputs/states suggested by the optimization problem). If replacing with the "max" operator we assume instead an adversarial or indifferent approach where the worst combination of inputs/states of the exterior agents has to be taken into account. □

Both the cost function (3.39a) and the feasible domain (3.39b)–(3.39c) depend explicitly upon the values found in the sets $\mathcal{U}_{\mathcal{I}\setminus\mathcal{N}_i}$ and $\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}$ which characterize the behavior (input and state) of the agents exterior to the current neighborhood. Consequently, the content of these sets can accommodate a large span of distributed control strategies. If no information is forthcoming from the exterior, then these sets are defined using reachable analysis, thus resulting in a decentralized control. At the other extreme, when the exact state of the exterior agents is communicated we have a distributed cooperative approach.

Usually, distributed approaches necessitate several iterations in-between consecutive discretization steps. That is, at discretization step k for a state $x(k)_i$ we may have $\bar{p}$ iterations with the intermediate values x_i^p where $p = 0 : \bar{p}$ and $x_i^0 \leftarrow x(k)$ and $x(k+1)_i \leftarrow x_i^{\bar{p}}$. The computations stop after some predefined number of iterations or when a consensus is reached.

Here we propose a hierarchical implementation which avoids by construction consensus verification and requires only one iteration. To this end we consider a hierarchical ordering of the neighborhoods[8]. For neighborhood $\mathcal{N}_i$, the remaining indices, represented by $\mathcal{I}\setminus\mathcal{N}_i$, are partitioned into $\mathcal{N}_i^-$ and $\mathcal{N}_i^+$ which denote the neighborhoods with lower, respectively higher, priority. Then, the sets $\mathcal{U}_{\mathcal{I}\setminus\mathcal{N}_i}$ and $\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}$ are constructed as follows:

$$\mathcal{U}_{\mathcal{I}\setminus\mathcal{N}_i} = \left(\bigcup_{j\in\mathcal{N}_i^-}\{u_j^0\}\right) \cup \left(\bigcup_{j\in\mathcal{N}_i^+}\{u_j^1\}\right), \tag{3.41a}$$

$$\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i} = \left(\bigcup_{j\in\mathcal{N}_i^-}\{x_j^0\}\right) \cup \left(\bigcup_{j\in\mathcal{N}_i^+}\{x_j^1\}\right), \tag{3.41b}$$

where superscript "0" and "1" denote the current iteration for a certain variable (e.g., u_j^0 means that the state is not yet updated and the initial value $u(k)_j$ will be used, whereas u_j^1 means that we use the updated value, the one which will define the next discretization step, $u(k+1)_j$).

Under the aforementioned constructive assumptions we state the following lemma dealing with constraints verification.

[8]Note that in the hierarchical implementation the neighborhoods are disjoint, see also the definition of neighborhoods $\mathcal{N}_i$ in Section 3.

Lemma 3.1. *Let the agents be in a feasible agent formation at discretization step k:*

$$\mathbf{x}(k) \in \mathcal{C}(\mathbb{S}). \tag{3.42}$$

By applying optimization problems (3.39) with sets (3.41) for each of the neighborhoods $\mathcal{N}_i$ which partition $\mathcal{I}$, we preserve the formation feasibility at the next discretization step $k+1$:

$$\mathbf{x}(k+1) \in \mathcal{C}(\mathbb{S}). \tag{3.43}$$

□

Proof. We proceed by induction. Consider neighborhood $\mathcal{N}_i$ and its attached optimization problem (3.39). Due to the construction of sets $\mathcal{U}_{\mathcal{I}\setminus\mathcal{N}_i}$ and $\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}$ as in (3.41), the optimization problem "sees" the agents of higher order in their updated positions and the ones of lower order in their initial positions. Then, the resulting control $u_{\mathcal{N}_i}(k+1)$ will lead to a state $x_{\mathcal{N}_i}(k+1)$ which respects the new states of the higher order agents (since they are explicitly included in the constraint set) and the un-updated states of the lower-order agents. On the other hand, the lower-order agents will not break the constraints involving agents with index in $\mathcal{N}_i$ because from their point of view, this neighborhood has a superior position in the hierarchy.

The presence of the constraints associated to the lower order agents guarantees the feasibility of $\mathbf{x}(k+1)$. At the i^{th} optimization problem, the initial state of the lower order agents is respected, thus, if no movement is possible for them, they can at lest keep the same state. This final argument completes the global recursive feasibility proof of the control scheme. □

Remark 3.9. As mentioned in the proof of Lemma 3.1 we consider for lower order agents the un-updated state and thus we guarantee the existence of a solution (at worse, the lower priority agent will be able to keep the same state[9]). This approach can be generalized by assuming not a single point (issued from the prediction) is provided as information to the lower priority neighborhoods, but rather all the points that can be reached by that agents with higher priority. In other words, this means that agents situated lower in the hierarchy could be "pushed-around" within acceptable bounds in a robust control manner. □

Remark 3.10. Note that the partitioning between neighborhoods needs not to be time invariant. If we consider that the neighborhoods have a geometric meaning (for example the indices of agents which are physically close) it may be necessary to change their content at every (few) discretization steps. In this sense, we point toward the k-means clustering algorithms, which permit partitioning a collection of agents into a predefined

[9]Not necessarily true when the dynamics describe systems which have a minimal velocity – unmanned aerial vehicles (UAVs) for example. However, in the present result, such limitations on the control action are not taken into account and the theoretical results holds.

number of groupings and partitions the state space into Voronoi cells [Kanungo et al., 2002]. □

To clarify the exposition we provide in Algorithm 3.2 a sketch of the distributed control problem.

Algorithm 3.2: Distributed MPC scheme

Input: obstacles $\{\mathcal{O}_o\}$, safety regions $\mathcal{S}_i$, neighborhoods $\mathbb{N}$, initial inputs and states $\mathcal{U}_{\mathcal{I}\setminus\mathcal{N}_i}, \mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}$, agent dynamics (A_i, B_i) and global cost function $V(\cdot,\cdot)$

-describe the feasible cells C_l (3.31) partitioning the feasible region defined by the collision avoidance conditions (3.16) as in Section 2.5, Chapter 2;
for $k = 1 : k_{max}$ **do**
  foreach $\mathcal{N}_i \in \mathbb{N}$ **do**
    -for neighborhood $\mathcal{N}_i$ calculate $\mathcal{U}_{\mathcal{I}\setminus\mathcal{N}_i}$ and $\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}$ as in (3.41);
    -construct the feasible region $\mathcal{C}(\mathbb{S})|_{\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}}$ as in Proposition 3.1;
    -write $\mathcal{C}(\mathbb{S})|_{\mathcal{X}_{\mathcal{I}\setminus\mathcal{N}_i}}$ in MIP formulation as in Section 2.5, Chapter 2;
    -solve optimization problem (3.39);
  end
  - $k = k + 1$;
end

3.5.3 Exemplification of a hierarchical distributed approach

For illustrative purposes let us consider the following example. Consider 3 agents, each one of them its own neighborhood: $\mathcal{N}_i = \{i\}$, $i = 1 : 3$. We order these neighborhoods lexicographically, that is, $\mathcal{N}_1 < \mathcal{N}_2 < \mathcal{N}_3$ in the hierarchical point of view and apply the optimization procedure (3.39). For clarity of the exposition we keep a one-step MPC problem (i.e., $N_p = 1$), such that only one step-ahead has to be considered in the constraints.

In Figure 3.12 we consider the 3 agents and show their evolution. Note that the first and last frames represent the current step k and next discretization step $(k+1)$ respectively, and that the 3 intermediate frames represent the single iteration executed in-between the discretization steps (inter-sample negociation). Further, we detail the execution of this iteration. In the second frame, we solve optimization (3.39) for $\mathcal{N}_1$. Since this neighborhood is the highest in the ordering ($\mathcal{N}_1^+ = \emptyset$ and $\mathcal{N}_1^- = \{2, 3\}$) all the other agents are kept in their initial position and agent 1 positions itself as depicted by the dashed red contour. At the next frame, the third, we solve the optimization problem for $\mathcal{N}_2$ to which correspond $\mathcal{N}_2^+ = \{1\}$ and $\mathcal{N}_2^- = \{3\}$. In this case, $\mathcal{N}_2^+$ is not empty and thus the agent's 1 updated state is used and agent 3 has its initial state. It can be seen that agent 2 finds a better state which respects both the updated and the

initial constraints. The same procedure is repeated at the next frame for $\mathcal{N}_3$ to which correspond $\mathcal{N}_3^+ = \{1, 2\}$ and $\mathcal{N}_3^- = \emptyset$. In the last frame it can be seen that each of the agents has changed its position and that the constraints were respected.

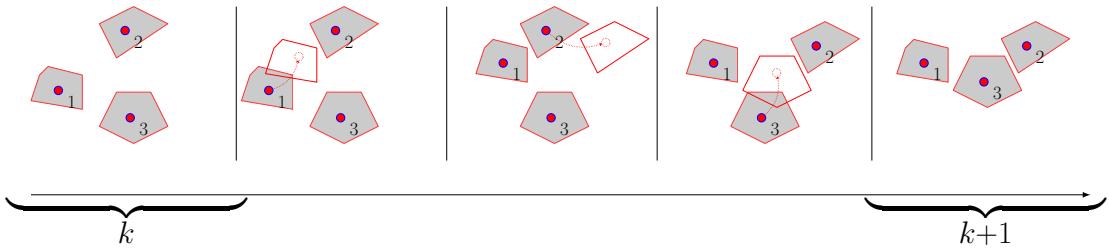


FIGURE 3.12: Exemplification of the hierarchical distributed approach.

As mention in the introduction of the present chapter, various control methods for solving the collision avoidance problem are related to the potential field approach [Tanner et al., 2007], graph theory [Lafferriere et al., 2004] or other optimization-based approaches which handle indirectly the constraints by penalty terms in the cost function. For example, the potential field-based methods offer an arguably less complex implementation with a smaller computational effort. In the following we investigate such a decentralized control strategy which is a combination of MPC and Potential Field approach as an alternative for the explicit collision avoidance approach presented up to this point. Beside the comparison value, the main contribution of the following work is oriented towards the construction of polyhedral potential functions and the decrease of the computational complexity since the MIP formulation is inherently difficult to solve.

3.6 Decentralized MPC

As seen in the preceding section, a main motivation for changing the optimization problem is the computational difficulty of the centralized problem. A first step was to decompose the optimization problem (to "distribute" it), and solve iteratively the subproblems until a consensus is reached. Here, we go further and assume that the agents do not seek consensus, that is, they are decentralized ("everybody for itself"). To better exemplify the decentralized approach that we adopt, Figure 3.13 depicts the main idea. Therefore, each agent has an associate controller. The exchange of information is reduced to a minimum, that is, the controller may receive information from the agents neighborhoods but does not adapt its solution with respect to the other solution provided by the rest of the controllers. A very simple visualization would be: two agents reach for the same one-agent-wide entrance, in centralized (and in a well constructed distributed) approach, one of the agents will cede the priority and let the other pass first. For the decentralized case with limited communication and lack of global optimality, it will be entirely possible that both agents will try to pass simultaneously and block each other.

Even tough it lacks the theoretical guarantees characterizing the centralized and distributed approaches, in practice, the proposed decentralized method exhibits effective performances by avoiding the centralized design which can be computationally demanding[10]. As such it deserves a certain attention at least for understanding the advantages and the sensitive aspects.

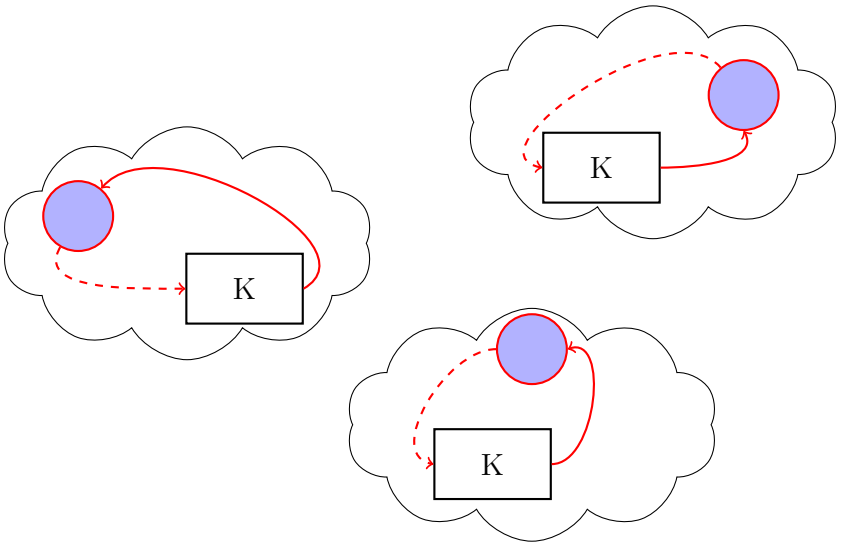


FIGURE 3.13: Decentralized MPC architecture.

3.6.1 MIP-based solution

3.6.1.1 A naive approach

The simplest (naive) approach when having a multi-agent system is to simply ignore the inter-agent interaction and obstacle collisions. Obviously, this will lead to a bad behavior for the system with irreversible damages. There are ways to improve the behavior while still not considering the non-convex constraints. The simplest one is to provide references (or some offsets) which will force the agents to stay away from each other. The problem with this approach is that the formation cannot be tight and that, any unpredictable disturbance can still lead to intersections of the agents' safety regions.

A partial consideration of the constraints, but without any consensus, may also help. For example, let's say that the agent looks around and constructs its trajectory accordingly.

[10]Besides computational expenses, the logistic difficulties can be mentioned: a centralized approach means that there exists a central processing unit which needs information from all the agents and has to send the control action to all agents. Such a construction may be difficult to implement and be prone to errors (e.g. the case of a radio-linked system of agents and a non-neglected physical obstacle which cuts the communication inside a subgroup).

Since its control action is based only on what it can "see" (position) and estimate (speed for example), we still do not have any guarantee of collision avoidance.

We have mentioned these two directions because even if they do not provide guarantees in practice they work reasonably well. However, what we want to stress is that, if collisions do not have disastrous consequences, such a scheme can be considered for implementation (consider the example of small robots playing football, since their speed is limited and they are robust from the point of view of mecatronic construction, eventual collisions will not destroy them).

3.6.1.2 Adjustment for a recursive collision avoidance

Here we go further and decompose the global system (3.27) of N_a agents into $\frac{N_a(N_a-1)}{2}$ subsystems of two agents, with similar objective function and state constraints (3.30). An optimization problem is associated with each subsystem and computes the local control inputs based only on the states of the agents and a subset of selected neighbors (with states available for measurements).

The set of neighbors of a given subsystem q can be defined as:

- $N_q = \{r \in \{1, \ldots, N_a\} : q\text{-th subsystem has at its disposal the measured state of the agents and has to impose a constraint involving the predicted state for the } r\text{-th subsystem.}$
- Each local MPC algorithm provides a sequence of control moves for the local subsystem and its neighbors.
- Only the first component of such control sequence is applied to the respective subsystem, no information about the control actions being exchanged at a global level.

Using the same structure as in the centralized case, for the discrete model of the q-th subsystem (and its neighbors), the local receding horizon optimization problem is the one used in the centralized case (3.30), while respecting the constraints imposed by the dynamics of the subsystem and the collision avoidance constraints defined as in Section 3.2. Due to the fact that a subsystem has restricted information about the global state and since the agents are in motion, the preservation of a relative distance between the agents of neighboring subsystems can not be guaranteed (mainly during transitory motion). This behavior imposes a modification of the constraints bounds leading to cooperative configurations similar with the one depicted in Figure 3.14, where the agents are ordered and the safety region of each agent is expanded (by homothety) according to its cardinal. For example, the first agent must be at a distance d from the second agent in each direction. The safety region of the first agent is expanded with d,

and imposes a $2d$ distance with respect to the $3^r d$ agent, which consequently implies a certain distance (d in this case) between the third and the second agent. We continue by expanding the safety region of the first agent with d up to $(N_a - 1)d$.

Such a control scheme can guarantee constraints-free behavior but the "tightness" of the formation is far from being optimal, the control strategy being over precautions. This conclusion can be in fact generalized: "in the collision free guaranteed strategies, the tightness of the formation has to be negotiated in terms of the communication capabilities".

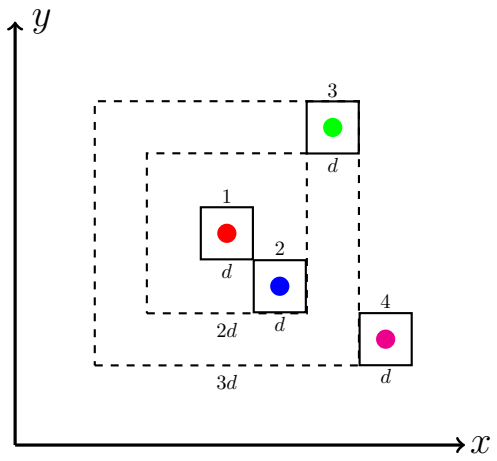


FIGURE 3.14: Cooperation constraints of the decentralized MIP-based solution.

3.6.2 Potential field-based solution

We choose here to combine two novel elements: the decentralized approach and the Potential field method, as an alternative to the usual MPC approach.

A different class of methods for collision avoidance problems uses artificial potential fields [Khatib, 1986] to directly obtain feedback control actions steering the agents over the entire workspace. One shortcoming of this approach is that the value of the gradient can be too large, which can enforce the possible generation of traps (local minima). Relevant research on generating so-called navigation functions that are free from local minima is available in the literature [Rimon and Koditschek, 1992]. However, generating a navigation function is computationally involved and thus not suitable for many navigation problems.

The solution we propose here is to consider the combined use of MPC and Potential field method in a decentralized implementation. More precisely, we add input constraints which will limit the gradient and by looking ahead on a certain number of prediction

steps we can minimize the chances of blocking into a local minima. We have in effect discarded the non-convex constraints appearing in the combined use of MPC and MIP and, relocated them into the potential field (the non-convex cost function of the MPC). The proposed solution is decentralized, the steering policy for each agent is based only on local state information from its nearest neighbors. Whenever the agent employs a continuous cycle of sensing and acting, a collision-free control law for the agent is computed in each cycle based on the local communication with its neighbors on a certain radius.

Until now we have presented solutions to a formation control problem where all the agents in the group had the same characteristics and goals. In the present section, we will deal with the decentralized control of a typical *leader-follower formation*. Here, the leader has the responsibility of guiding the group, while the followers have the responsibility of maintaining the inter-agent formation.

3.6.2.1 Decentralized system description

Consider the same set of N_a constrained systems, each individual nominal system being described by (3.3). We assume that the components of the state are[11]: the position $p^i(t)$ and the velocity $v_i(k)$ of the i^{th} agent such that $x_i(k) = [p_i^T(k)\ v_i^T(k)]^T$. For further use, we will make the difference between the leader agent and the followers by indexing it with $l \in \mathcal{I}$, where $\mathcal{I}$ defines the collection of all agents indices (3.1).

Let us assume the steering policy for each follower agent based only on local state information from its nearest neighbors.

Definition 3.2. *[Neighboring graph [Tanner et al., 2007]] An undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ represents the nearest neighboring relations and consists of:*

- *a set of vertices (nodes) $\mathcal{V} = \{n_1, n_2, \cdots, n_{N_a}\}$ indexed by the agents in the group;*
- *a set of edges $\mathcal{E} = \{(n_i, n_j) \in \mathcal{V} \times \mathcal{V} : n_i \leftrightarrow n_j\}$, containing unordered pairs of nodes that represent neighboring relations.*

If we simplify and consider the inter-agent distance, the single limiting factor for their communication we can define the observation neighborhood of agent $i \in \mathcal{I}$ and $i \neq l$ as follows:

$$\mathcal{N}_i(k) \triangleq \{j \in \mathcal{I} : \|p_i(k) - p_j(k)\| \leq r,\ i \neq j\}, \tag{3.44}$$

where r is the radius of the ball centered in p_i.

[11] We consider a particular form with respect to the previous sections with a specific structure for the state. This is required by the method we wish to employ: Potential Field approaches are usually considered for systems which have position and velocity as components of the state.

Remark 3.11. Note that in Definition 3.2 we introduce a new notion, which can be interpreted as "observation neighborhood", which can be different than the neighborhood notion described in Definition 3.1 for the distributed case. The first one has a physical meaning: it concentrates the agents, which are close enough, to be observed by the current agent. On the other hand, the neighborhoods, as they were defined in the distributed case (see Definition 3.1 in Section 3.5.1) stand for a grouping of agents whose control actions are given by the same optimization problem[12]. □

In a potential field approach the collision avoidance problem is treated by considering penalty terms in the cost function. Since in our approach the agents have associated safety regions, in the following we construct repulsive potential functions which take into account their shape. Note that the proposed polyhedral potential field description can be adapted to most of the existing control architectures and thus is generic.

3.6.2.2 Repulsive potential functions for collision avoidance

Our interest is mainly in taking into account the shape of the safety region (in terms of (3.9)) of an agent navigating by the obstacles and/or other moving entities. As we have seen in previous sections, the collision avoidance conditions are described by the equations (3.14)–(3.16), that is the intersection between the agents safety regions and/or obstacles needs to be void. As such, in the sequel we employ the use of the polyhedral function (2.26) and sum function (2.30) (see Section 2.4.2 and Section 2.4.3 from Chapter 2) for the construction of repulsive potential functions, which have a high value inside the convex safety regions and a progressively increasing value outside them.

In order to exemplify their influence in the collision avoidance problem, we propose several repulsive potential functions constructed through the use of the formulations (2.26) and (2.30).

First, the construction based on the polyhedral function defined in (2.26) is proposed for the generation of a repulsive potential:

$$V_\mu(\mu(x)) = c_1 e^{-(\mu(x)-c_2)^2}, \tag{3.45}$$

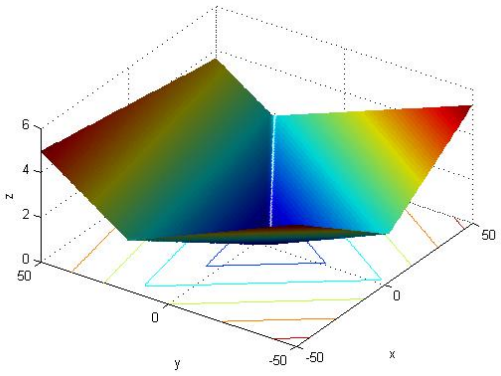
where the parameters c_1 and c_2 are positive constants representing the strength and effect ranges of the repulsive potential.

Second, an alternative repulsive potential using the sum function described in (2.30) is given by:

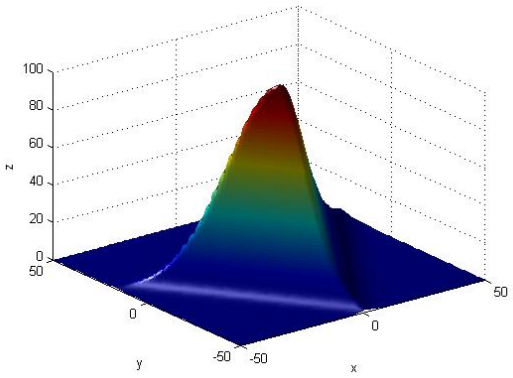
$$V_\psi(\psi(x)) = \frac{c_3}{(c_4 + \psi(x))^2}, \tag{3.46}$$

[12]Strictly speaking the notion could be used in the development of the results in the distributed control section but we have preferred to keep the problem simpler at that stage and thus we assumed an infinite range of communication.

with c_3 and c_4 positive constants representing the strength and effect ranges of the repulsive potential (3.46).

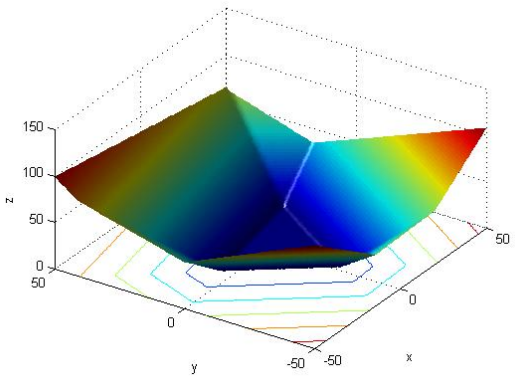


(A) Polyhedral function (2.30) of the bounded polyhedral set in Figure 2.4.

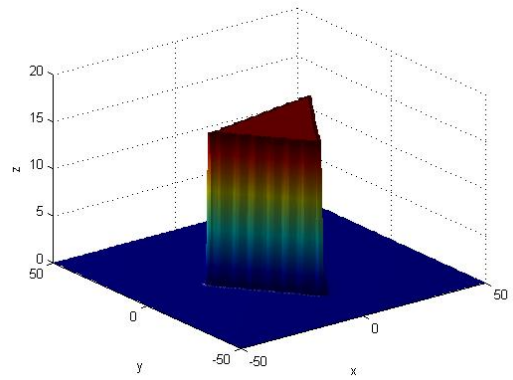


(B) Repulsive potential using the polyhedral function.

FIGURE 3.15: Repulsive potential construction using polyhedral function.



(A) Sum function (2.26) of the bounded polyhedral set in Figure 2.5.



(B) Repulsive potential using the sum function.

FIGURE 3.16: Repulsive potential construction using sum function.

As an illustration we recall here the polyhedral function depicted in Figure 2.4 (also depicted in Figure 3.15 (a)) and using (3.45) we construct the repulsive potential function depicted in Figure 3.15 (b). A similar construction is provided in Figure 3.16 by using the repulsive potential function (3.46). As it can be seen, both functions have a high value inside the polytopes and a low value outside them.

The repulsive potential will be further used in order to derive a control action such that the collision avoidance inside the formation is satisfied. Next, we proceed to the description of the decentralized optimization problem.

3.6.2.3 Decentralized optimization-based control problem

The proposed control approach will be applied for solving a typical leader-follower formation control problem. The goal is to coordinate the agents designed as followers to achieve a formation while following the agent designed as the leader.

The repulsive potential functions introduced in (3.45) and (3.46) produce a potential field. The negative gradient of this potential is the key element towards a collision free behavior for the agents. Globally, an attractive component of the potential function aims at maintaining a given formation. In this context, we provide a practical control design method which enables the decentralized decision making for a leader-followers group of agents. If not otherwise specified, we choose randomly (off-line) the leader from the available agents and solve a simple tracking problem for it. A finite receding horizon implementation of the optimal control law is typically based on the real-time construction of the control sequence $\mathbf{u}_l = \{u_l(k|k), u_l(k+1|k), \cdots, u_l(k+N_{p,l}-1|k)\}$ that minimizes the finite horizon quadratic objective function:

$$\mathbf{u}^* = \arg\min_{\mathbf{u}^l}(x_l(k+N_{p,l}|k)^T P x_l(k+N_{p,l}|k) + \sum_{s=1}^{N_{p,l}-1} x_l(k+s|k)^T Q x_l(k+s|k)+ \tag{3.47}$$

$$+ \sum_{s=0}^{N_{p,l}-1} u_l(k+s|k)^T R u_l(k+s|k)),$$

subject to:

$$\begin{cases} x_l(k+s+1|k) = A_l x_l(k+s|k) + B_l u_l(k+s|k), \quad s = 0, \ldots, N_{p,l}-1, \\ x_l(k+s|k) \in X_l, \quad s = 1, \ldots, N_{p,l}, \\ u_l(k+s|k) \in U_l, \quad s = 1, \ldots, N_{p,l}. \end{cases} \tag{3.48}$$

Note that the optimization problem for the leader is a restricted form of the optimization problem (3.30) in the case that the target position is the origin (which is admissible in the absence of the remaining agents). Here $Q = Q^T \succeq 0$, $R \succ 0$ are positive definite weighting matrices, $P = P^T \succeq 0$ defines the terminal cost and $N_{p,l}$ denotes the prediction horizon for the leader. The optimization problem (3.47) has to be solved subject to the dynamical constraints (3.48). In the same time, other security or performance specifications can be added to the system trajectory. The sets X_l, U_l have to take into account the reference tracking type of problem delineated in (3.47).

In the following, we concentrate our attention on the coordination of the rest of the agents denoted as followers (all the agents besides the leader). Since the agents are in motion, their relative distances can change with time, affecting their neighboring sets (3.44). For each agent i, we define an inter-agent potential function which aims to accomplish the following objectives:

1) collision avoidance between agents;

2) convergence to a group formation while following the leader.

More precisely, in the proposed problem, the following inter-agent potential function is used:

$$V_i(p_i, v_i) = \beta_r V_i^r(p_i) + \beta_a V_i^a(p_i, v_i),\ \forall i \in \mathcal{N}_i. \tag{3.49}$$

The two components of the potential function account for the objectives presented above and β_r, β_a are weighting coefficients for each objective. For the i^{th} agent the total potential is formed by summing the potentials terms corresponding to each of its neighbors. Consequently, in the present approach, the potential functions are designed as follows:

1) $V_i^r(p_i)$ denotes the repulsive potentials that agent i sense from its neighbors:

$$V_i^r(p_i) = \sum_{j \in \mathcal{N}_i} V_{ij}^r(p_i) \tag{3.50}$$

To implement this, the concepts introduced in Section 3.6.2.2, specifically the potential functions (3.45) or (3.46) are taken into account:

$$V_{ij}^r(p_i) = \frac{c_3}{(c_4 + \psi_{ij}(p_i))^2},\ i \neq j,\ i \neq l, \tag{3.51}$$

where $\psi_{ij}(p_i)$ is the sum function (2.30) (see, Section 2.4.3, Chapter 2) induced by the the safety regions (3.9) associated to both the followers and the leader.

2) $V_i^a(p_i, v_i)$ denotes the attractive component between agents in order to achieve a formation and to follow the leader agent:

$$V_i^a(p_i, v_i) = \sum_{j \in \mathcal{N}_i} V_{ij}^a(p_i, v_i) + \|p_l - p_i\|, \tag{3.52}$$

for all $i \in \mathcal{N}_i$ and $i \neq l$.
The second component denotes the relative distance between the leader and the followers. The first component $V_{ij}^a(x_i)$ has the following form:

$$V_{ij}^a(p_i, v_i) = \log(\psi_{ij}^2(p_i)) + \beta_v(v_i - v_j), \tag{3.53}$$

where β_v denotes a weighting coefficient for which the agents velocities are synchronizing.

Note that the issue of choosing appropriate weights in the above relations (3.49), (3.51), (3.53) is of greatest importance and highly depends on the model (inertia, maneuverability, disturbance levels etc), generally following the basic rules of energy versus tracking performance compromise, their computational aspects being beyond the scope of this work. In the implementation of the present approach we choose empirical values which satisfied our requirements, for the convergence purpose, the potential function being required to have a unique minimum when agents i and j are located at a desired distance. In practice, we usually choose values for the weights c_1, c_2, c_3, c_4 of the repulsive potentials (3.45)–(3.46) in a range between 0.1 and 1 as well as for the weight β_r in (3.49). Furthermore, for the weights β_a and β_v of the attractive potentials (3.49)–(3.53) we choose higher values, between 10 and 20. Similar works based on potential filed methods provide insightful discussions on the tuning of the parameters. Among these papers we cite [Wu et al., 2010], [Barnes et al., 2009] where a procedure to set the parameters to accomplish convergence of multi-agents systems is presented.

Note also that for a potential function, a piecewise affine gradient can be computed (see the gradient of the polyhedral and sum functions in Chapter 2). As in [Rimon and Koditschek, 1992], [Tanner et al., 2007] the negative value of the gradient can be applied in order to derive a control action for agent i. The direct approach has several shortcomings mentioned in the Section 3.6.

In the following, we formulate the optimization problem (3.47) for the followers, by using the potential-based cost function described in (3.49). A control sequence $\mathbf{u}_i = \{u_i(k|k), u_i(k+1|k), \cdots, u_i(k+N_p-1|k)\}$ which minimizes the finite horizon nonlinear objective function:

$$\mathbf{u}^* = \underset{\mathbf{u}_i}{\arg\min} \left(\sum_{s=0}^{N_p} V_i(p_i(k+s|k), v_i(k+s|k)) \right). \tag{3.54}$$

subject to:

$$\begin{cases} x_i(k+s+1|k) = A_i x^l(k+s|k) + B_i u^l(k+s|k), \ s = 0, \ldots, N_{p,f}-1, \\ x_i(k+s|k) \in X_f, \quad s = 1, \ldots, N_{p,f}, \\ u_i(k+s|k) \in U_f, \quad s = 1, \ldots, N_{p,f}. \end{cases} \tag{3.55}$$

Here $N_{p,f}$ denotes the prediction horizon for the followers. The sets X_f, U_f denote in a compact formulation the magnitude constraints on states and inputs.

In the optimization problem (3.54) we need to know the future values of the neighboring graph and the values of the state for the corresponding neighbors. All these elements are time-varying and difficult to estimate. For computational reasons the following assumptions will be made:

- The neighboring graph is considered to be constant along the prediction horizon, that is,

$$\mathcal{N}_i(k+s|k) \triangleq \mathcal{N}_i(k). \tag{3.56}$$

- The future values of the followers state are considered constant

$$x_j(k+s|k) \triangleq x_j(k). \tag{3.57}$$

- An estimation of the leader's state is provided by the equation [13]

$$x_l(k+s|k) \triangleq 0. \tag{3.58}$$

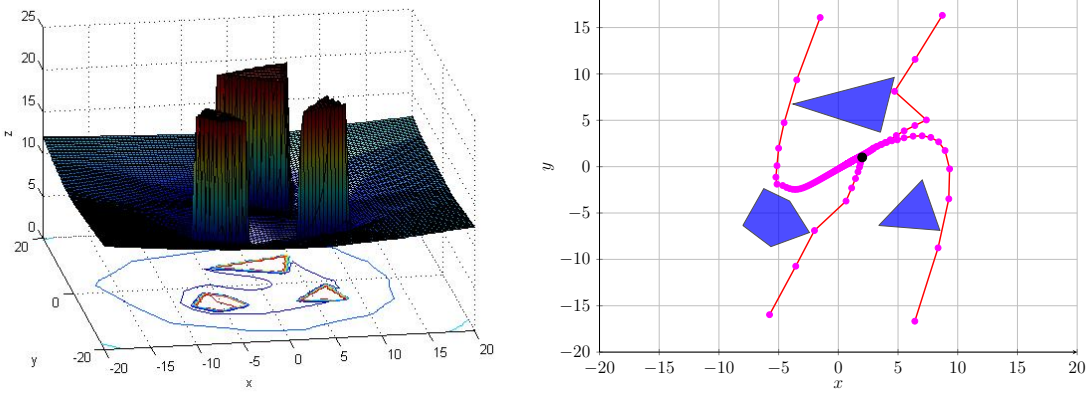
The equations (3.56)–(3.58) represent only rough approximation of the future state of the agents. Obviously, the MPC formulation can be improved by using prediction of the future state of the neighboring agents (leading to a min-max MPC strategy). Where feasible, the conservatism of the prediction may be improved by the agents themselves via communication of their optimization results [Dunbar and Murray, 2006]. Here a simplified approach was implemented for the followers (by assuming constant predictions) and using the reference trajectory for the leader.

Remark 3.12. The time-varying nature of the neighboring graph and the fact that the future values of the neighboring states and the leader state are unknown represent some of the computational limitations of the presented scheme. Moreover, the resulting cost function is nonlinear and, more than that, non-convex. This means that the numerical solution may suffer from the hardware limitations and may not correspond to the global optimum. □

We want also to mention that the stability issues are in some sense still open and this particular work has a formal contribution principally towards the use of polyhedral functions and only secondly towards their integration in the global control strategy. Furthermore, the introduction of the invariant safety region in the repulsive part of the potential function limits the collision occurrences. Recall that for the collision avoidance problem, the nominal dynamics of the agent is used, but it is require that the safety region of the real agents do not intersect. The constraints being "soft" means that we have no absolute guarantee of collision avoidance but in practice, we observe that such constraints are respected.

Before exemplifying the Potential Field-based formation control approach, let us validate the use of the proposed repulsive potential functions in an obstacle avoidance example.

[13] The leader state is estimated to be 0 since it does not have a reference trajectory to follow, otherwise $x_l(k+s|k) = \tilde{x}_l^{ref}(k+s)$, where $\tilde{x}_l^{ref}(k+s)$ may represent the reference trajectory followed by the leader agent.



(A) Potential filed in a workspace with 3 obstacles.

(B) Agent trajectories using predictive control.

FIGURE 3.17: Collision avoidance using repulsive potential functions.

Exemplification for obstacle avoidance using repulsive potentials

Let us consider a punctiform agent (i.e. $N_a = 1$ in (3.10) with $m_1 = 45$kg, $\mu_1 = 15$Ns/m) operating in a two spatial dimensions environment with three obstacles defined by (3.9) (represented by the blue convex regions in Figure 3.17). We consider a potential function defined by (3.49) ($\beta_r = 1$, $\beta_a = 1$), with two components: a repulsive potential (3.46) with $c_3 = 1$, $c_4 = 0.25$ and an attractive potential (3.53). The potential function generates a potential field depicted in Figure 3.17 (a). First, we calculate the gradient of the potential function which is piecewise affine as in (2.31), Section 2.4.3, Chapter 2. The negative value of the gradient is applied in order to derive a control action for the agent. We obtain that the obstacles are usually avoided, with a reduce rate of violation (90% in the presence of random noise in the aditive disturbance). There are, however, situations when the constraints are not satisfied, or the control action obtained through the negative gradient has unrealistic values. For these reasons, we introduce the potential function in a predictive control framework as in (3.54) leading to a constrained free MPC, with a prediction horizon $N_{p,f} = 2$. We obtain that the obstacles are always avoided. Figure 3.17 (b) illustrates several trajectories of the agent with a random initial position.

Exemplification for the decentralized navigation of multi-agent formation

Consider five homogeneous agents (i.e. $N_a = 5$ in (3.10)) described by the dynamics (3.10), with $m_1 = 45$kg, $\mu_1 = 15$Ns/m. The initial positions and velocities of the agents are chosen randomly. We take arbitrarily $l = 1$ to be the leader which has to be followed

by the rest of the agents $i = 2, \ldots, 5$ ($i \neq l$). A quadratic cost function as defined in (3.47) is used for the leader over a prediction horizon $N_{p,l} = 10$.

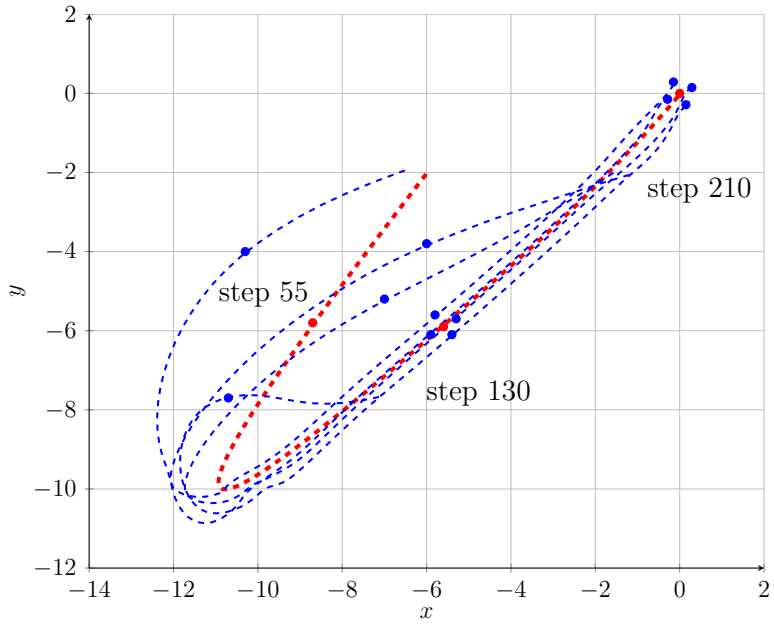


FIGURE 3.18: Evolution of a leader/followers formation at different time instances with their safety regions (leader in red, followers in magenta).

For the followers we consider a potential function as the cost function in the optimization problem (3.54), with a prediction horizon $N_{p,f} = 2$. The potential will be constructed such that both the tracking of the leader and the maintaining of a formation are respected. The neighborhood radius is set to $r = 8$m, the weighting coefficients are $\beta_r = 1$, $\beta_a = 10$, $c_3 = 1$, $c_4 = 0.25$, $\beta_v = 15$. The effectiveness of the present algorithm is confirmed by the simulation depicted in Figure 3.18, where the evolution of the agents is represented at three different time instances. The agents successfully reach a formation and follow the leader without colliding.

We note that we prefer a smaller prediction horizon for the followers than the one used for the leader. This is justified by the fact that the trajectory of the leader has a higher priority and that any additional prediction step for the potential function (which is not quadratic) incurs significant computational complexity.

3.7 Concluding remarks

In this chapter, we first present several tools in order to provide a systematic off-line procedure for the control of a group of agents towards a minimal configuration. Secondly,

in real-time a two stage receding horizon control design is adopted for driving the agents to the predefined formation. For the convergence to the predefined formation is shown that an additional fixed point constraint (i.e., the target positions are also equilibrium points for the considered dynamics) must be taken into account.

The Mixed-Integer Programming (MIP) technique presented in the previous chapter enables the geometric description of the feasible region resulting from the typical multi-agent collision and obstacle avoidance constraints.

The formation control problem is also discussed from a centralized, distributed and decentralized point of view. Each of these approaches has different strengths and weaknesses. The centralized MPC implementation offers the best theoretical guarantees for multi-agent formation stability but it is numerically cumbersome.

On the other hand, the set of agents can be portioned into neighborhoods and a distributed predictive control problem can be solved. Here we have concentrated on the geometrical aspects of the feasible region and preferred a simple hierarchical strategy. Thus, we avoid inter-sample iterations which seek consensus. While quite flexible, this scheme raises interesting and challenging questions. For example, it is not yet clear how to efficiently implement a non-hierarchical distributed approach which would make efficient use of the MIP constructions. Further, set-theoretic elements can be added and we estimate that they can significantly improve upon the scheme (e.g., the use of reachable sets to limit the feasible domain).

As a last element, we discuss a decentralized approach and show that, even with the lack of theoretical guarantees of stability, the simulations results exhibit a functioning within parameters. Also here we analyze an alternative to the MIP construction used until now. The MIP formulation is inherently difficult to solve, as such, in the las part of the present chapter we choose the Potential Field methodology as alternative design where the constraints are no longer “hard” but rather “soft”, in the sense that we impose penalties in crossing them in the cost function. The solution is decentralized, the steering policy for each agent is based only on local state information from its nearest neighbors. We emphasize that our interest in using potential field approach was mainly in taking into account the shape of safety region of an agent navigating by the obstacles and/or other moving entities. The proposed constructions use repulsive potentials based on special class of (symmetrical) piecewise linear functionals. More precisely, here we use both the sum function and polyhedral function in order to provide constructions for the repulsive potential.

Chapter 4

Assessments of the constrained control of multi-agent dynamical systems

The previous chapters highlight the importance of collision avoidance in multi-agent control problems. This fundamental problem turns out to be difficult, one of the main reasons being the non-convexity of the resulting feasible region.

As already detailed along the first part of the thesis, there is a wealth of methods for formulating and solving the multi-agent optimization problem. The common and dominating factor of all these methods is the presence, in a form or another, of non-convex constraints. They are used explicitly in the MIP formulations and implicitly in the Potential Field formulations, but no matter the approach, they are always present from the design stage. What we want to emphasize is that, non-convex constraints are not just an artifact of the problem, but rather an intrinsic property which cannot be avoided (as it was a long tradition with convex constraints via anti-windup strategies for example). Solving problems over non-convex regions is not a new issue in the literature (at least from the optimization point of view), but in the present chapter we fix as objective to go further in the analysis of the limit behavior and the selection of appropriate orbit for the closed-loop trajectories. The first remark in this sense is that, the natural unconstrained equilibrium point (the origin in the case of LTI dynamics) becomes infeasible. More precisely, the type of constraints that we consider makes the convergence of an agent dynamics towards such an equilibrium point impossible to fulfill.

In the literature, the problem of the avoidance of convex fixed obstacles is examined in [Rakovic et al., 2007] from a set-theoretic point of view. The authors propose to deal with the non-convex control problem by considering approximation procedures which "inner and outer convexify" the exact capture sets. Furthermore, in [Raković and Mayne, 2007]

the same problem is tackled by using set computations and polyhedral algebra. Other work, [Patel and Goulart, 2011] reports a gradient-based optimization algorithm for trajectory generation for aircraft avoidance maneuvers where the obstacles are defined as convex regions. However, in all of these papers the origin is part of the feasible region. To the best of the authors knowledge, there does not exists any results treating the case of constraints which are not satisfied by the origin (understood as the equilibrium point of the dynamical system to be controlled).

The first part of the present chapter is twofold. In a first stage, we perform a detailed analysis of the limit behavior for a *linear* dynamical system in the presence of *geometric adversary constraints*. More precisely, we need to define the fixed points and the invariance properties for the system state trajectory while avoiding a convex region containing the origin in its strict interior. In the context of multi-agent systems, this region can, in fact, represent an obstacle (static constraints) but can be extended to the safety region of a different agent (leading to a parameterization of the set of constraints with respect to the global current state).

In a second stage, our interest is to ensure the stability over the feasible region of the state space using a dual-mode strategy. The basic principles are those of Model Predictive Control (MPC) technique including avoidance constraints. There is a fundamental difference to the classical MPC which rely on the assumption that the origin is in the relative interior of the feasible region (see, for example, [Mayne et al., 2005], [Seron et al., 2000], [Bemporad et al., 2002]) or on the frontier of the feasible region [Pannocchia et al., 2003].

The links between fixed points, invariance and affine feedback laws allow us to provide *necessary and sufficient conditions* for the existence of a stable equilibrium point having the entire feasible region as a basin of attraction.

Alternatively, similar conditions can be formulated in order to guarantee that the closed-loop system is unstable over the entire feasible region. These conditions can be subsequently used for the design of unstable control laws [Chetaev, 1952] with the ultimate goal of "repelling" the trajectories from a certain *sensitive* region of the state space. However, these considerations are out of the main topic of the present work. Hereafter, we concentrate on the algebraic conditions for excluding limit cycles. The employed methods are specific to Piecewise Affine (PWA) systems analysis, with a geometric insight on the invariance properties of polytopic regions in the state space.

In last part of the present chapter, we go further and extend the analysis for a multi-agent formation. The previous fixed point uniqueness condition translate in the multi-agent formulation into a unique configuration condition. It is important to assure that a control action will not lead to a cycling behavior for the agents, which implies energy consumption. Formally, the fact that a set of agents remains in (or arrives to) a unique and stable configuration, as a result of a suitable control strategy, is equivalent with saying that in an extended space, there exists and it is unique a point which can be

made fixed through the same control strategy. Therefore, we can guarantee the stability of a multi-agent formation, but of course, that it depends on the way we solve the optimization problem. At the end of the chapter, we recall the optimization-based strategies presented in Chapter 3 and provide discussions based on simulation results on whether a stable formation can be achieved.

Finally, there are many applications of the present work which are of particular interest, namely those where static or dynamic constraints must be respected. Examples include coordinated ocean platform control for a mobile offshore base [Girard et al., 2001]. The homogeneous modules forming the offshore base must be able to perform long-term station keeping at sea, in the presence of waves, winds and currents. Therefore, the independent modules have to be controlled in order to be maintained aligned. This task can be easily accomplished if each module converges to different fixed points, which are suitably chosen. Furthermore, it is well known that the North Atlantic is one of the most inhospitable environments on the planet. Yet, it is here that Atlantic Norway's offshore oil and gas industry (see, [Grant and Shaw, 2001], [Bertino and Lisæter, 2008]) has been operating for years and with an impressive amount of success. In order to remain prosperous under such harsh conditions, the offshore industry relies on a variety of innovative technologies. An example is the ice-breaking cargo vessel and/or tanker which need to break the ice around the platform. Therefore, the ice-breaking vessel has to maneuver as close as possible to the platform, while avoiding the collision with it (i.e. the vessel has to converge to a limit cycle).

4.1 Preliminaries

4.1.1 A generic problem formulation

Let us recall the system (2.9) and express its dynamics in a simplified form:

$$x(k+1) = Ax(k) + Bu(k), \tag{4.1}$$

where $x(k) \in \mathbb{R}^n$ is the state, $u(k) \in \mathbb{R}^m$ is the input signal and A, B are state matrices of appropriate dimensions. It is assumed that the pair (A, B) is *stabilizable*.

The minimization of infinite horizon cost function (usually a quadratic function involving states and inputs) leads to the linear state-feedback control law characterizing the optimal *unconstrained* control:

$$u(k) = K_{LQ}x(k), \tag{4.2}$$

with K_{LQ} computed from the solution of the discrete algebraic Riccati equation.

Consider now the problem of optimal control of the system state (4.1) towards the origin while its trajectory *avoids* the polyhedral region defined as in (2.17) (Section 2.4.1, Chapter 2) and recalled here:

$$S = \left\{ x \in \mathbb{R}^n : h_i^T x < k_i, \quad i = 1, \dots, N \right\}, \tag{4.3}$$

with $(h_i, k_i) \in \mathbb{R}^n \times \mathbb{R}$ and N being the number of half-spaces. Recall also that we focus on the case where $k_i > 0$ for all $i = 1, \dots, N$, meaning that the origin is contained in the strict interior of the polytopic region, i.e. $0 \in S$. Note that, the feasible region is a non-convex set defined as the complement of (4.3), namely $\mathbb{R}^n \setminus S$. From this simple problem formulation the inconsistency between the two requirements becomes obvious. The origin is infeasible but represents the convergence point for the closed-loop trajectories.

The first implication of this contradictory facts is that, whenever (4.2) is infeasible, a corrective control action needs to be taken such that the system's trajectories remain outside the prohibited region (4.3):

$$x(k) \notin S. \tag{4.4}$$

A tractable approach is the recursive construction of an optimal control sequence $\mathbf{u} = \{u_{k|k}, u_{k+1|k}, \cdots, u_{k+N_p-1|k}\}$ over a *finite* constrained receding horizon, which leads to a predictive control policy:

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} (x_{k+N_p|k}^T P x_{k+N_p|k} + \sum_{s=1}^{N_p-1} x_{k+s|k}^T Q x_{k+s|k} + \sum_{s=0}^{N_p-1} u_{k+s|k}^T R u_{k+s|k}), \tag{4.5}$$

subject to the set of constraints:

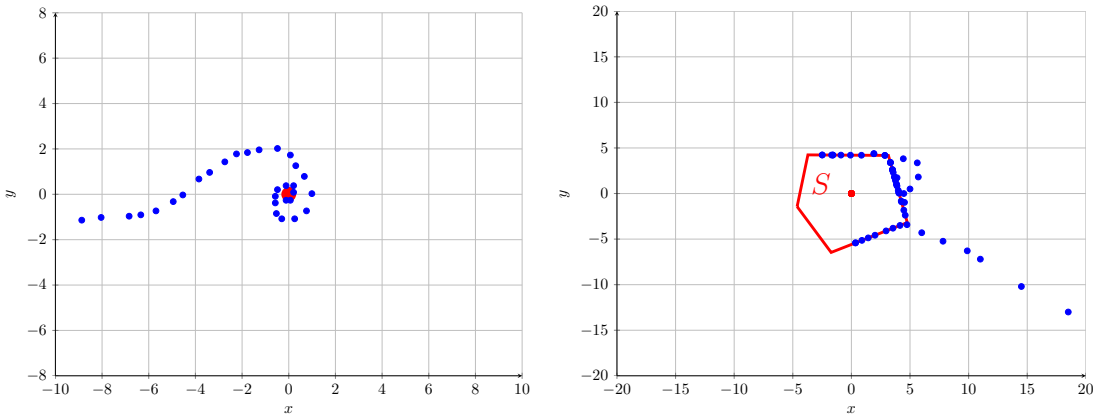
$$\begin{cases} x_{k+s+1|k} = A x_{k+s|k} + B u_{k+s|k}, & s = 0, \dots, N_p - 1, \\ x_{k+s|k} \in \mathbb{R}^n \setminus S, & s = 1, \dots, N_p. \end{cases} \tag{4.6}$$

Here $Q = Q^T \succeq 0$, $R \succ 0$ are weighting matrices, $P = P^T \succeq 0$ defines the terminal cost and N_p denotes the length of the prediction horizon.

Illustrative example

In order to understand our motivation in analyzing the behavior of an agent when adversary constraints are imposed we show in Figure 4.1 (a) the trajectory of a linear agent when unconstrained optimum (4.2) is applied. Observe Figure 4.1 (b), the chaotic behavior of the agent trajectory, when (4.5) is applied subject to constraints (4.6).

In the sequel, a first insight on the geometry of such a problem with a single constraint will be discussed. Due to the tractable formulation of the explicit solution a one-step



(A) Agent trajectory using unconstrained optimum (4.2).

(B) Polytope S and system trajectory by solving the constrained optimization problem (4.5).

FIGURE 4.1: System trajectories with and withought geometric adversary constraints.

ahead prediction will be employed in order to show the particularities of the closed-loop dynamics.

4.1.2 An explicit solution for a particular case

For a straightforward understanding of the issues of limit behavior in the presence of adversary constraints let us consider the a one step ahead MPC problem and $S = \left\{x \in \mathbb{R}^2 : h_1^T x < 1\right\}$. Briefly, the LQ optimal control action is admitted as long as the trajectory does not pass through the constraints. Otherwise, the control action is modified such that the constraint is activated. Figure 4.2 describes the regions obtained through partitioning with the imposed constraints, the hyperplane of the restricted region and the explicit MPC partitioning, respectively.

The optimization problem to be solved is formulated as follows:

$$u^*(k) = \arg\min_{u(k)} \left(x(k+1)^T P x(k+1) + u(k)^T R u(k)\right), \tag{4.7}$$

$$\text{subject to: } \begin{cases} x(k+1) = Ax(k) + Bu(k) \\ h_1^T x(k+1) \geq 1 \end{cases}$$

After introducing $x(k+1)$ in the cost function and the inequality constraint, the optimization problem (4.7) is reformulated as follows:

$$u^*(k) = \underset{u(k)}{\arg\min} \left(\frac{1}{2} u^T(k)(R + B^T P B)u(k) + x^T(k) A^T P B u(k)\right), \tag{4.8}$$

$$\text{subject to: } -h_1^T B u(k) \leq -1 + h_1^T A x(k).$$

The solution of (4.8) is a continuous piecewise affine function of x, defined over a polyhedral partition $\{L_j\}_{j\in\{0,1\}}$ of $L \subseteq \mathbb{R}^2$ satisfying $L = \bigcup_{j\in\{0,1\}} L_j$ and for which $L_0 \cap L_1$ have a relative empty intersection. Explicitly, the solution of (4.8) is defined as:

$$u^*(k) = F_j x(k) + G_j, \quad \forall\ x(k) \in L_j \text{ and } j \in \{0, 1\}, \tag{4.9}$$

where $F_j^T \in \mathbb{R}^2$, $G_j \in \mathbb{R}$.

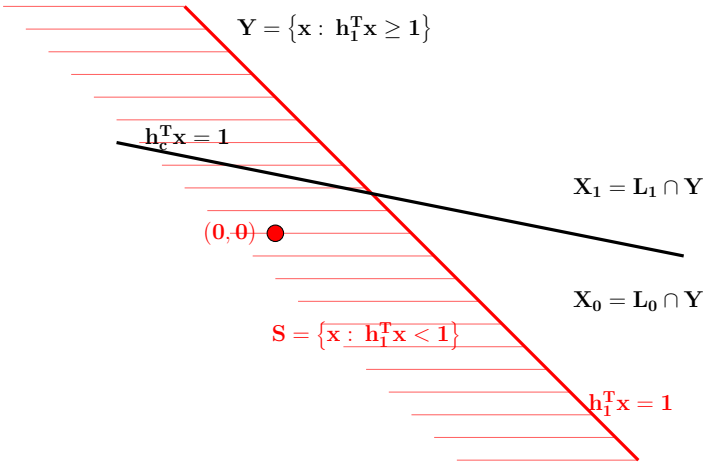


FIGURE 4.2: Exemplification of the regions determined by the unbounded adversary constraints.

As illustrated in Figure 4.2, the polyhedral partitions $\{L_j\}_{j\in\{0,1\}}$ are defined as:

$$L_0 = \{x \in \mathbb{R}^2 : h_c^T x \geq 1\}, \tag{4.10}$$

$$L_1 = \{x \in \mathbb{R}^2 : h_c^T x < 1\}, \tag{4.11}$$

where $h_c \in \mathbb{R}^2$ is given by the first-order Karush-Kuhn-Tucker (KKT) optimality conditions [Bemporad et al., 2002]:

$$h_c^T = h_1^T (A - B \underbrace{(R + B^T P B)^{-1} B^T P^T A}_{K_{LQ}}). \tag{4.12}$$

Let us consider

$$Y = \left\{ x \in \mathbb{R}^2 : h_1^T x \geq 1 \right\}, \tag{4.13}$$

where $0 \notin Y$ (from the definition of the restricted region (4.3), for the case $N = 1$). Define X_0, $X_1 \subseteq \mathbb{R}^2$ such that, $X_0 = L_0 \cap Y$ denotes all the state trajectories from W which under the LQ dynamics (4.2) remain in W at the next iteration and $X_1 = L_1 \cap W$ denotes all the state trajectories that at the next iteration transit in the restricted region. The previous definitions can be formally described as:

$$X_0 = \{x \in \mathbb{R}^2 : h_1^T x \geq 1,\ h_1^T (A + BK_{LQ})x \geq 1\}, \tag{4.14}$$

$$X_1 = \{x \in \mathbb{R}^2 : h_1^T x \geq 1,\ h_1^T (A + BK_{LQ})x \leq 1\} \tag{4.15}$$

with $X_0 \cup X_1 = Y$.

Remark 4.1. One of the optimal solutions (4.9) corresponds to the unconstrained optimum (4.2). Without a loss of generality it is assumed to be for the partition L_0 (i.e. the affine term is $G_0 = 0$). □

Therefore the closed-loop system can be described by the following piecewise affine dynamics:

$$x(k+1) = \begin{cases} (A + BK_{LQ})x(k), & \text{for } x(k) \in X_0 \\ (A + BF_1)x(k) + BG_1, & \text{for } x(k) \in X_1 \end{cases} \tag{4.16}$$

Remark 4.2. The existence of the constraint in the optimization problem (4.8) makes the origin, which represents the fixed point for the dynamics associated to the polyhedral set X_0, to reside outside the validity domain: $0 \notin X_0$. □

Proposition 4.1. *Consider the system (4.1) with the control law defined by (4.9). The following statements are verified:*

a. *All the state trajectories initiated in $x(k) \in X_0$ transit in a finite time to X_1.*

b. *The dynamics associated to the polyhedral set X_1 defined by (4.16) will steer any point from X_1 in one step on the boundary of the feasible region, i.e.* $\{x \in \mathbb{R}^2 : h_1^T x = 1\}$.

□

Proof: See Appendix. □

Remark 4.3. Note that a consequence of Proposition 4.1.b is that one of the eigenvalues of the state matrix associated to X_1 is 0 since the dynamics of X_1 represent a projection on a $n - 1$ dimensional set.

Let us concentrate on simple mathematical conditions for the existence and uniqueness of a stable fixed point on the frontier of the feasible domain. The qualitative behavior

of the agent (4.1) is determined by the pattern of its fixed points, as well as by their stability properties. One issue of practical importance is whether the fixed point x_e associated to the polyhedral region X_1 is stable or unstable and whether x_e is included in X_1 or X_0.

The following theorem is stated as simple sufficient condition for the existence and uniqueness of a stable/attractive fixed point of system (4.16).

Theorem 4.1. *Consider the discrete-time system* (4.1) *in closed-loop, with the optimal solution* (4.9) *of the optimization problem* (4.7).

If there exists $V \in \mathbb{R}^{2\times 2}$ *with nonnegative elements such that*

$$\begin{bmatrix} -h_1^T \\ h_c^T \end{bmatrix} (A + BF_1) = V \begin{bmatrix} -h_1^T \\ h_c^T \end{bmatrix}, \tag{4.17}$$

and

$$(V - I)\mathbf{1} < -\begin{bmatrix} -h_1^T \\ h_c^T \end{bmatrix} BG_1, \tag{4.18}$$

with h_c *defined by* (4.12) *and*
$F_1 = -G_1 h_1^T h_c - (R + B^T P B)^{-1} B^T P A$,
$G_1 = (R + B^T P B)^{-1} B^T h_1 (h_1^T B (R + B^T P B)^{-1} B^T h_1)^{-1}$,
then the following properties with respect to $x_e = (I - (A + BF_1))^{-1} BG_1$ *hold:*

a. *If* $x_e \in X_1$, *then* x_e *is a unique stable fixed point with a basin of attraction* Y *defined by* (4.13).

b. *If* $x_e \notin X_1$, *then the closed-loop dynamics are globally unstable in* Y *with* x_e *an unstable fixed point. Moreover, all the trajectories transit to infinity along the boundary of the feasible region.*

□

Proof: See Appendix. □

4.1.3 Further geometrical insights for the generic MPC problem

We revert in this section to the problem statement in (4.5)–(4.6) and analyze via explicit solutions the geometry of the closed-loop system in the case of bounded convex set S as in (4.3) which needs to be avoided.

Proposition 4.2. *Let the pair* (A, B) *in* (4.1) *be controllable, the set* S *as in* (4.3) *be nondegenerate and bounded with* $0 \in S$ *and* $A + BK_{LQ}$ *nonsingular. Then the closed-loop system* $x(k+1) = Ax(k) + B\boldsymbol{u}_1^*(k)$, *where* $\boldsymbol{u}_1^*(k)$ *is the first component of the optimal sequence* u^* *in* (4.5), *is defined by a PWA dynamics with the following properties:*

1. *The partition of the state-space covers* $\mathcal{C}(S) = \mathbb{R}^n \setminus S$. *More than that,* $\mathcal{C}(S)$ *is unbounded in any direction of* $\mathbb{R}^n$.

2. *The PWA dynamics over* $\mathcal{C}(S)$ *contains a finite number of polyhedral regions.*

3. *There is a convex region* $\mathcal{R}$ *such that over* $\mathbb{R}^n \setminus \mathcal{R}$ *the optimal control action* $\boldsymbol{u}_1^*(k) = K_{LQ}x(k)$ *according to* (4.2).

4. *All the initial states in* $\mathbb{R}^n \setminus \mathcal{R}$ *will converge to* $\mathcal{R}$ *in finite time.*

□

Proof: In order to prove the properties let us stress the structure of the optimization problem in (4.5)–(4.6). This can be cast in the class of multiparametric problems with $x(k)$ as a vector of parameters. Moreover, it can be reformulated as a *mixed-integer multiparametric quadratic program* where a binary vector of decision variables is used to code the feasible region in (4.6).

1) The first property is related to the boundedness of the set S and the controllability properties of the pair (A, B) defininining the dynamics in (3.6). We need to show that for any point in $\mathcal{C}(S)$ there exists a control action such that the constraints are satisfied. Consider then the first prediction step: from the controllability assumptions we have that for any point $x_0 \in \mathcal{C}(S)$ there exists a control action $u_0 \neq 0$ such that $(Ax_0 + Bu_0) \neq 0$. Moreover, there exists a scalar $\alpha \neq 1$ such that $(1-\alpha)Bu_0 \neq 0$ and thus, by the boundedness of the set S, $(Ax_0 + B\alpha u_0) \notin S$. By induction the same holds for all the steps along the prediction horizon and the feasibility of $\mathcal{C}(S)$ is proved.

2) The second property follows directly for the structural properties of the multiparametric optimization problems. Indeed, the solution will be piecewise affine and the partition of the feasible set will be upper bounded from the point of view of the number of regions by the possible combinations of active sets. As long as (4.6) has a finite number of constraints (finite prediction horizon) a finite number of regions in the PWA partition [Pistikopoulos et al., 2007].

3) We will provide here a constructive proof. Consider the set $\mathcal{R}$ as:

$$\mathcal{R} = ConvHull(S, (A + BK_{LQ})^{-1}S, \ldots, (A + BK_{LQ})^{-N_p}S), \tag{4.19}$$

where N_p is the length of the prediction horizon. For any point in $x_0 \in \mathbb{R}^n \setminus \mathcal{R}$ the unconstrained optimal sequence

$$u_f^* = \{K_{LQ}x_0, K_{LQ}(A + BK_{LQ})x_0, \ldots, K_{LQ}(A + BK_{LQ})^{N_p-1}x_0\} \tag{4.20}$$

is feasible as it is not part of the sequence which generated the set $\mathcal{R}$. The consequence is that u^* do not saturate any constraint in (4.6) and thus the optimal solution correspond

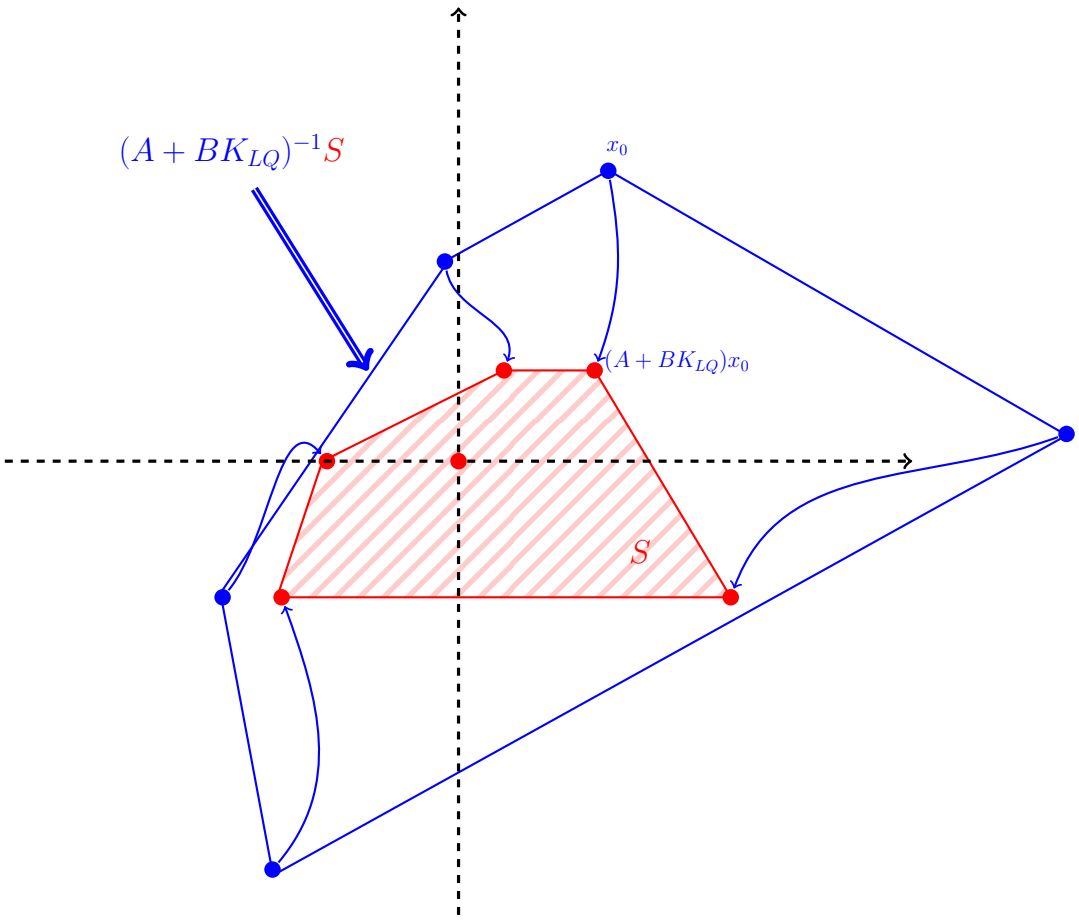


FIGURE 4.3: Prohibited region S and one-step reachable set corresponding to set $\mathcal{R}$.

to the unconstrained optimum $\mathbf{u}^* = u_f^*$. The fact that $u_1^*(k) = K_{LQ}x(k)$ completes the proof (see an illustration in Figure 4.3).

4) The previous property shown that over $\mathbb{R}^n \setminus \mathcal{R}$ the closed-loop dynamics correspond to the LTI system $x(k+1) = (A + BK_{LQ})x(k)$ which is known to be asymptotically stable. Moreover, the set $\mathcal{R}$ is bounded and there is a ball $\mathbb{B}_\delta \subset \mathcal{R}$. Using the $\epsilon - \delta$ stability arguments one can prove that for any initial state in $x_0 \in \mathbb{R}^n \setminus \mathcal{R}$ there exists a finite number number of steps l such that $x_l \in \mathbb{B}_\delta \subset \mathcal{R}$. □

Remark 4.4. The properties enumerated in the above result hold also by relaxing the singularity assumption on the $A + BK_{LQ}$ matrix. However the proof of the third item is slightly more involved as it will need the recursive construction of a reachable set. For the sake of simplicity we resume the exposition to the argumentation of the nonsingular case. □

The fact that the closed loop dynamics exhibit a PWA dynamics, lead us to the conclusion that the stability analysis can be performed via sufficient Piecewise Quadratic Lyapunov arguments as in [Hovd and Olaru, 2010]. However, these tools can be employed under the assumption of the existence of a unique equilibrium point which is not guaranteed here. In fact, number of PWA analysis problems as for example deciding on the number and the basin of attraction of the orbits of a PWA system is known to be a NP-hard problem [Asarin et al., 2000].

Furthermore, the Proposition 4.2 proves the existence of a compact region $\mathcal{R}$ with an associated piecewise affine map. The region $\mathcal{R}$ is positive invariant and by simple scaling one can assure besides the invariance, the contractivness of this set. Thus, we are in the case of a structurally stable map and the celebrated Smale's Horseshoe example is there to show that such mapping can have infinitely many periodic orbits and the set $\mathcal{R}$ can be even a *chaotic invariant set* [Wiggins, 1988].

Besides satisfying the constrains (4.6), additionally we would like that the systems' state approaches a unique equilibrium point and avoids the existence of multiple basins of attraction, cyclic or chaotic behavior. In the general case the periodic solutions can be considered as optimal candidates for the limit behavior. In the present work, the control objective is to avoid limit cycles and concentrate on the convergence to a unique fixed point with basin of attraction $\mathcal{C}(S) = \mathbb{R}^n \setminus S$.

Remark 4.5. Usual MPC concerns regarding *feasibility* or *recursive feasibility* are not critical for the problem (4.5)–(4.6) as long as the feasible set is actually unbounded. Such discussion becomes relevant if additional *convex* state/input constraints are to be handled. We refer to reachability studies in order to deal with these problems which are out of the scope of the present chapter. □

4.2 Local constrained control

In the previous subsections it was shown that the generic MPC formulation can lead to closed-loop behavior which is difficult to analyze locally around the region to be avoided (even by explicit MPC formulations). Hence, a local-to-global approach should be preferred for the control design. In this section, we first establish conditions for an affine state-feedback control law to render a half-space positively invariant. Second, we associate the half-space to an equilibrium state lying on its boundary. Then, these conditions are used for the derivation of a control law that transfers the system's state as close as possible to the origin, while avoiding the prohibited region.

4.2.1 Equilibrium states

Consider an affine control law of the form:

$$u(k) = K(x(k) - x_e) + u_e, \tag{4.21}$$

with $x_e \in \mathbb{R}^n$ the desired equilibrium state, $K \in \mathbb{R}^{m \times n}$ a generic feedback-gain matrix and $u_e \in \mathbb{R}^m$ the feed-forward parameter. The resulting closed-loop system is described by the state equation

$$x(k+1) = Ax(k) + BK(x(k) - x_e) + Bu_e, \tag{4.22}$$

and $x(k) - x_e$ defines its transient behavior.

A state x_e is an equilibrium state for the closed-loop system (4.1) if:

$$x_e = Ax_e + Bu_e. \tag{4.23}$$

Lemma 4.1. *The points x_e represent equilibrium states for the dynamics (4.1) if and only if it belongs to the preimage through the linear map $(I_n - A)$, of the linear subspace, spanned by the columns of matrix B.* □

Proof: The vector u_e in (4.23) can be seen as a degree of freedom in the generation of the points x_e in (4.23). As such, the columns of B spans a subspace which is further mapped through the linear application $(I_n - A)$. □

Remark 4.6. The geometrical locus of the equilibrium states is independent of K. Consequently, the states that can be equilibria are defined by the dynamics of the unforced system and are completely specified by u_e. □

Illustrative example

As previously mentioned, the set of equilibrium points $x_e \in \mathbb{R}^n$ is the image of matrix $(I_n - A)^{-1}B$ in the case when $(I_n - A)$ is non-singular. The particular characteristics of the dynamics (state and input dimensions) define the shape of the subspace of states that can be equilibria. Figure 4.4 depicts a 2-dimensional system with a scalar input. It can be seen that the geometrical locus of the fixed points is in fact, a line which passes through the origin.

4.2.2 Positive invariance conditions

We can further concentrate on one of the key issues for the control design: the controlled invariance with respect to an affine control law (4.21) and subsequently, the closed-loop stability. For solving this problem, the following lemma provides algebraic invariance

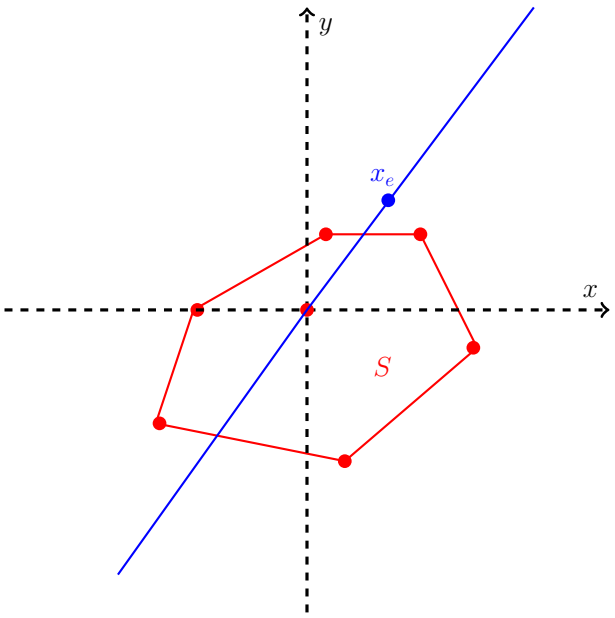


FIGURE 4.4: Prohibited region and geometrical locus of equilibrium states.

conditions. Note that this result is a particular case of a more general result established in [Bitsoris and Truffet, 2011].

Lemma 4.2. *The half-space defined by the inequality $v^T x \leq \gamma$ is a positively invariant set of the affine system $x(k+1) = Mx(k) + c$, if and only if there exists a positive real number g such that:*

$$v^T M = g v^T, \tag{4.24}$$

and

$$g\gamma + v^T c \leq \gamma. \tag{4.25}$$

□

Remark 4.7. From Lemma 4.2, it follows that

$$v^T M = g v^T, \tag{4.26}$$

and

$$-g\gamma - v^T c \leq -\gamma, \tag{4.27}$$

are necessary and sufficient conditions (obtained by the inversion of the signs with respect to the linear constraint) for the opposite half-spaces defined by inequality $v^T x \geq \gamma$ to be positively invariant with respect to system $x(k+1) = Mx(k) + c$. Note however that, (4.24)–(4.25) is not equivalent to (4.26)–(4.27) and thus, the two half-spaces need to be treated with the appropriate conditions (mainly through the use of the appropriate affine term in the affine control law (4.21)). □

The next result exploits the invariance properties and relates the above algebraic conditions to the corresponding equilibrium states.

Theorem 4.2. *If $x_e \in \mathbb{R}^n$ is an equilibrium state of the closed-loop system (4.1) lying on the hyperplane $v^T x = \gamma$, then a necessary and sufficient condition for this hyperplane to partition the state-space into two positively invariant half-spaces is that $v^T \in \mathbb{R}^{1\times n}$ to be* left *eigenvector of the closed-loop matrix $A + BK \in \mathbb{R}^{n\times n}$ associated to a positive eigenvalue $\lambda \neq 1$.* □

Proof: Applying Lemma 4.2 to system (4.22) leads to:

$$x(k+1) = (A + BK)x(k) + B(u_e - Kx_e) \tag{4.28}$$

and considering Remark 4.7 the invariance yields the following algebraic conditions:

$$v^T(A + BK) = \lambda v^T, \tag{4.29}$$
$$\lambda\gamma + v^T B(u_e - Kx_e) \leq \gamma, \tag{4.30}$$
$$-\lambda\gamma - v^T B(u_e - Kx_e) \leq -\gamma, \tag{4.31}$$
$$\lambda \geq 0. \tag{4.32}$$

Equation (4.29) directly proves that (λ, v^T) is a (eigenvalue/*left* eigenvector) pair. From (4.30) and (4.31) we have

$$\lambda\gamma + v^T B(u_e - Kx_e) = \gamma, \tag{4.33}$$

which can be rewritten as:

$$\lambda\gamma + v^T((A + BK)x_e + B(u_e - Kx_e)) - v^T(A + BK)x_e = \gamma. \tag{4.34}$$

Then, relation (4.34) becomes $\lambda\gamma + v^T x_e - \lambda v^T x_e = \gamma$ or $(1-\lambda)v^T x_e = (1-\lambda)\gamma$. Considering the hypothesis $\lambda \neq 1$, we obtain $v^T x_e = \gamma$, thus proving the sufficiency.

Conversely, the invariance of the half-space $v^T x \leq \gamma$ is equivalent to:

$$v^T(A + BK) = \lambda v^T, \tag{4.35}$$
$$\lambda\gamma + v^T B(u_e - Kx_e) \leq \gamma. \tag{4.36}$$

Condition (4.35) is satisfied by the eigenstructure properties. In the same time, x_e is an equilibrium state of the closed-loop system (4.28) lying on the hyperplane $v^T x = \gamma$. Then, x_e satisfies relation (4.23) and $v^T x_e = \gamma$. Exploiting these facts, condition (4.36) becomes $\lambda\gamma + v^T x_e - v^T(A + BK)x_e \leq \gamma$ and is equivalent to $\lambda\gamma + \gamma - \lambda v^T x_e \leq \gamma$. Finally, we have that $v^T x_e \geq \gamma$, which is trivially verified as long as $v^T x_e = \gamma$. Similar manipulations provides the invariance properties for the opposite half-space $v^T x \geq \gamma$ thus, proving the necessity. □

Illustrative example

The above results can be depicted for a 2-dimensional system $x_{t+1} = Mx(k) + c$ with scalar input. In Figure 4.5 the straight line which separates the state space into two invariant half-spaces and the trajectories converging to the equilibrium point are shown.

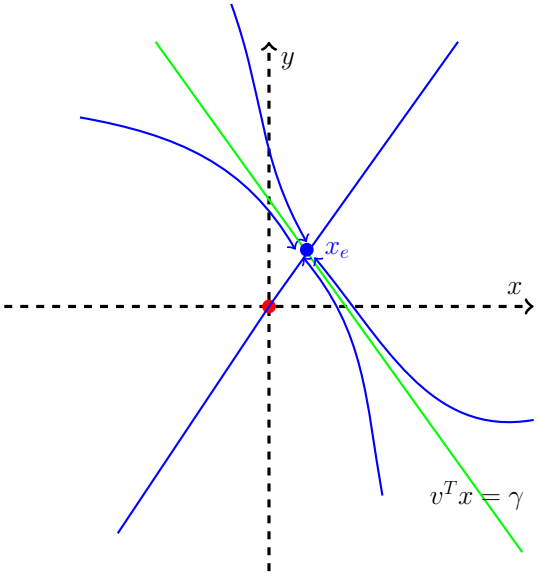


FIGURE 4.5: Invariant half-spaces for an affine system

4.2.3 Eigenstructure assignment analysis

As seen above, the eigenvector of a closed-loop matrix can be seen as the normal to a hyperplane. Under mild assumptions (related principally to the controllability), this hyperplane can partition the space into two complementary and invariant half-spaces. In the context of control design we are interested in the converse problem: *Given a hyperplane, does there exist a certain structural constraint on the gain matrix K which makes the resulting closed-loop matrix to have the normal to the hyperplane as an eigenvector? If not, which is the closest approximation possible (in the sense of the infinity norm)?*

These questions lead to an eigenstructure assignment analysis. Starting with the set of hyperplanes defining the polyhedral prohibited region (4.3), we search for the gain matrices of the control laws which assure stability and assign a left eigenvector as *close* as possible to the normal to a frontier of S in (4.3). Additionally, we show this to be optimal for some a priori given cost matrices (e.g., as in a Riccati equation setting).

The forthcoming results builds upon the remarks in [Srinathkumar and Rhoten, 1975], [Andry et al., 1983] and similar papers which detail the eigenstructure design in the early 80s. In particular, for a controllable systems it is said that:

- n eigenvalues and a maximum of $n \times m$ eigenvector entries can be arbitrarily specified,
- no more than m entries of an eigenvector can be chosen arbitrarily.

In the following, we will derive necessary and sufficient conditions for the existence of a gain matrix K which assigns a prescribed eigenvector and the associated eigenvalue.

Theorem 4.3. *Given a controllable pair* (A, B) *and the pair* $(\lambda, v) \in \mathbb{R} \times \mathbb{R}^n$ *the following relations hold:*

1. *If* (λ, v^T) *is the eigenvalue/*left *eigenvector pair associated to the matrix* $A + BK$*, then* $K \in \mathbb{R}^{m \times n}$ *satisfies*
$$K^T z = w, \tag{4.37}$$
with $w = (\lambda I_n - A)^T v \in \mathbb{R}^n$ *and* $z = B^T v \in \mathbb{R}^m$.
2. *If* (λ, v) *is the eigenvalue/*right *eigenvector pair associated to the matrix* $A + BK$*, then* $K \in \mathbb{R}^{m \times n}$ *satisfies*
$$Kv = \tilde{w}, \tag{4.38}$$
with $\tilde{w} \in \mathbb{R}^m$ *the solution of* $(A - \lambda I_n)v = -B\tilde{w}$.

□

Proof: 1) For the dynamics described by (4.1) and a given vector $v \in \mathbb{R}^n$, under controllability assumptions, there exists a matrix $K \in \mathbb{R}^{m \times n}$ such that the pair (λ, v^T) is an eigenvalue/*left* eigenvector of matrix $(A+BK)$ and K verifies the linear constraint:

$$v^T B \cdot K = w^T, \tag{4.39}$$

with $w \in \mathbb{R}^n$ and $w^T \triangleq v^T(\lambda I_n - A)$. The equation (4.37) is obtained from (4.39) by considering $z = B^T v$ under full-column rank hypothesis, concerning the matrix B.

2) Similarly, let $K \in \mathbb{R}^{m \times n}$ such that the pair (λ, v) is an eigenvalue/*right* eigenvector of matrix $(A + BK)$:

$$(A + BK)v = \lambda v. \tag{4.40}$$

If we rewrite (4.40) in the form $(A - \lambda I_n)v = -BKv$ we obtain the linear constraint (4.38) with $\tilde{w} \in \mathbb{R}^m$, a solution of the system of equations: $(A - \lambda I_n)v = -B\tilde{w}$. □

Remark 4.8. Note that Theorem 4.3 stresses a structural constraint on the gain K which depends on the placement of a *single pair* (eigenvalue/eigenvector). No assumption or particular implications are made with regards to the rest of the closed-loop matrix A+BK eigenstructure. Rather, the structural constraints (4.37) or (4.38) will be added further in a design procedure, to construct a gain which ensures, additionally to a particular invariance property, the stability of the unconstrained closed-loop system. □

The aforementioned theorem offers the framework for the following design procedure.

Description: An optimization problem can be formulated in order to find a *stable* eigenvalue associated to a *left/right* eigenvector which corresponds to a normal of a given hyperplane v. Concomitantly, a vector of parameters $w \in \mathbb{R}^n$ or $\tilde{w} \in \mathbb{R}^m$ and implicitly the linear structural constraint on the feedback gain K as in (4.37) or (4.38) is obtained.

Input: The controllable pair (A, B) describing the system (4.1) and the normal vector $v \in \mathbb{R}^n$ to a given hyperplane.

Output: A structural (linear) constraint (4.37) or (4.38) on the feedback gain ensuring the invariance property and the stability of the respective eigenvalue.

1.

$$\begin{aligned} &\min_{\delta,\lambda,w} \delta && (4.41)\\ \text{s.t.:} \quad &-\mathbf{1}\delta \leq v^T(A - \lambda I_n) + w^T \leq \mathbf{1}\delta \\ &\delta \geq 0, \\ &0 < \lambda < 1. \end{aligned}$$

2.

$$\begin{aligned} &\min_{\epsilon,\lambda,\tilde{w}} \epsilon && (4.42)\\ \text{s.t.:} \quad &-\mathbf{1}\epsilon \leq (A - \lambda I_n)v + B\tilde{w} \leq \mathbf{1}\epsilon, \\ &\epsilon \geq 0, && (4.43)\\ &0 < \lambda < 1. && (4.44) \end{aligned}$$

In the case when the optimal solution of the optimization problem (4.41) is $\delta^* = 0$, the vector v^T can be used for the separation of invariant half-spaces (as detailed in Lemma 4.2 and Theorem 4.2) with respect to the closed-loop dynamics. Moreover, the conditions imposed on the associated eigenvalue assure the contractiveness of the respective eigenvector.

The optimal argument $w^* \in \mathbb{R}^n$ of the LP problem (4.41) or $\tilde{w}^* \in \mathbb{R}^m$ of the LP problem (4.42), will be instrumental in the control design problem through a structural constraint on the fixed gain matrix as in (4.37) or (4.38).

4.2.4 Affine parametrization of the feedback policies

As it can be seen from the eigenstructure assignment approach described above, the main difficulty for proving the stability in the neighborhood of x_e is imposed by the structural constraint on the gain matrix inherited from the invariance desideratum. This imposes a reformulation of the local control problem in order to identify the design parameters. In the following, we will derive an affine parametrization of the feedback policies such that a fixed gain matrix K can be used for feedback, while respecting the constraint (4.37) or (4.38).

Theorem 4.4. *Consider the stabilization of system (4.1).*

1. *A stabilizing feedback gain K satisfying (4.37) exists if and only if the pair $(A + B\Gamma^T, B\Psi^T)$ is stabilizable with $\Gamma \in \mathbb{R}^{n\times m}$ and $\Psi \in \mathbb{R}^{(m-1)\times n}$ defined as:*

$$\begin{aligned} \Gamma &= \begin{bmatrix} \boldsymbol{0}_{n\times(m-1)} & w\tilde{z}^{-1} \end{bmatrix}, \\ \Psi &= \begin{bmatrix} \boldsymbol{I}_{(m-1)} & -\hat{z}\tilde{z}^{-1} \end{bmatrix}, \end{aligned} \tag{4.45}$$

   with $z = \begin{bmatrix} \hat{z} & \tilde{z} \end{bmatrix}$, $\tilde{z} \in \mathbb{R}$, $\tilde{z} \neq 0$, $\hat{z} \in \mathbb{R}^{m-1}$ and $w \in \mathbb{R}^n$ as in (4.37).

2. *A stabilizing feedback gain K satisfying (4.38) exists if and only if the following system is output stabilizable (through $u(k) = \breve{K}y(k)$):*

$$\begin{aligned} x(k+1) &= \left(A + B\tilde{\Gamma}\right) x(k) + Bu(k), \\ y(k) &= \tilde{\Psi}x(k), \end{aligned} \tag{4.46}$$

   with $\tilde{\Gamma} \in \mathbb{R}^{m\times n}$ and $\tilde{\Psi} \in \mathbb{R}^{(n-1)\times n}$ defined as:

$$\begin{aligned} \tilde{\Gamma} &= \begin{bmatrix} \boldsymbol{0}_{m\times(n-1)} & \tilde{w}\tilde{v}^{-1} \end{bmatrix}, \\ \tilde{\Psi} &= \begin{bmatrix} \boldsymbol{I}_{(n-1)} & -\hat{v}\tilde{v}^{-1} \end{bmatrix}, \end{aligned} \tag{4.47}$$

   with $v = \begin{bmatrix} \hat{v} & \tilde{v} \end{bmatrix}$, $\tilde{v} \in \mathbb{R}$, $\tilde{v} \neq 0$, $\hat{v} \in \mathbb{R}^{n-1}$ and $\tilde{w} \in \mathbb{R}^m$ as in (4.38).

□

Proof: 1) We start by decomposing $z \in \mathbb{R}^m$ in (4.37) into two elements $z = \begin{bmatrix} \hat{z} & \tilde{z} \end{bmatrix}$ such that the element $\tilde{z} \in \mathbb{R}$ (a non-zero scalar) and $\hat{z} \in \mathbb{R}^{m-1}$. Then, decomposing $K^T \in \mathbb{R}^{n\times m}$ similarly into $\hat{K}^T \in \mathbb{R}^{n\times 1}$ and $\tilde{K}^T \in \mathbb{R}^{n\times(m-1)}$ we can express after simple

algebraic manipulations $\hat{K}^T$ as a function of w, $\hat{z}$ and $\tilde{z}^{-1}$. Furthermore, introducing this into the original equality (4.37) we obtain an affine relation with K^T:

$$K^T = \Gamma + \tilde{K}^T \cdot \Psi \tag{4.48}$$

with $\Gamma \in \mathbb{R}^{n\times m}$ and $\Psi \in \mathbb{R}^{(m-1)\times n}$ defined as in (4.45). Using the above parametrization, relation (4.48) can be introduced into the closed-loop matrix as follows:

$$A + BK = A + B\left(\Gamma^T + \Psi^T\tilde{K}\right) = \left(A + B\Gamma^T\right) + B\Psi^T\tilde{K}. \tag{4.49}$$

This leads to a transformation of the original dynamics (4.1):

$$x(k+1) = \left(A + B\Gamma^T\right)x(k) + B\Psi^T\tilde{K}x(k) \tag{4.50}$$

and complete the equivalence between the original constrained stabilization problem and the controllability of the pair $(A + B\Gamma^T, B\Psi^T)$.

2) Similarly, $v \in \mathbb{R}^n$ in (4.38) can be decomposed into two elements $v = \begin{bmatrix}\hat{v} & \tilde{v}\end{bmatrix}$ such that the element $\tilde{v} \in \mathbb{R}$ (a non-zero scalar) and $\hat{v} \in \mathbb{R}^{n-1}$. As in the previous case, we obtain an affine description of $K \in \mathbb{R}^{m\times n}$ using the independent parameters contained in $\breve{K} \in \mathbb{R}^{m\times(n-1)}$:

$$K = \tilde{\Gamma} + \breve{K} \cdot \tilde{\Psi}, \tag{4.51}$$

with $\tilde{\Gamma} \in \mathbb{R}^{m\times n}$ and $\tilde{\Psi} \in \mathbb{R}^{(n-1)\times n}$ defined in (4.47). Using the parametrization (4.51) we obtain:

$$A + BK = A + B\left(\tilde{\Gamma} + \breve{K} \cdot \tilde{\Psi}\right) = \left(A + B\tilde{\Gamma}\right) + B\breve{K}\tilde{\Psi}. \tag{4.52}$$

This leads to a transformation of the original dynamics (4.1) into a novel formulation as described in (4.46), which is similar to an output feedback control problem. This highlights one of the important structural properties for the local constrained design: the equivalence of the stabilization and invariance in presence of active constraints with a specific output feedback design. □

Remark 4.9. Note that for $m = 1$ in (4.48), the gain matrix is directly imposed by $\Gamma = w\tilde{z}^{-1}$ since for this particular case the subspace defining $\tilde{K}$ is null. The same remark can be extended for $n = 1$ in (4.51), where the gain matrix is given by $\tilde{\Gamma} = \tilde{w}\tilde{v}^{-1}$. □

Illustrative example

We propose here an illustrative example of the reasoning leading to equations (4.48)–(4.45) (note that is similar in the case of (4.51)–(4.47)) and the subsequent values of the matrices involved. In this sense, let us consider a matrix $K \in \mathbb{R}^{2\times 2}$ which respects the

constraint (4.37). Then, we can write:

$$\begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix},$$

which is equivalent to:

$$\underbrace{\begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}}_{K^T \in \mathbb{R}^{2\times 2}} = \underbrace{\begin{bmatrix} 0 & w_1 z_2^{-1} \\ 0 & w_2 z_2^{-1} \end{bmatrix}}_{\Gamma \in \mathbb{R}^{2\times 2}} + \underbrace{\begin{bmatrix} k_{11} \\ k_{21} \end{bmatrix}}_{\tilde{K}^T \in \mathbb{R}^2} \underbrace{\begin{bmatrix} 1 & -z_1 z_2^{-1} \end{bmatrix}}_{\Psi \in \mathbb{R}^{1\times 2}}.$$

where each of the vectors/matrices corresponds with the notation in (4.48)–(4.45). A similar decomposition can be applied to another particular case, i.e. for $K \in \mathbb{R}^{3\times 3}$:

$$\underbrace{\begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix}}_{K^T \in \mathbb{R}^{3\times 3}} = \underbrace{\begin{bmatrix} 0 & 0 & w_1 z_3^{-1} \\ 0 & 0 & w_2 z_3^{-1} \\ 0 & 0 & w_3 z_3^{-1} \end{bmatrix}}_{\Gamma \in \mathbb{R}^{3\times 3}} + \underbrace{\begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \\ k_{31} & k_{32} \end{bmatrix}}_{\tilde{K}^T \in \mathbb{R}^{3\times 2}} \underbrace{\begin{bmatrix} 1 & 0 & -z_1 z_3^{-1} \\ 0 & 1 & -z_2 z_3^{-1} \end{bmatrix}}_{\Psi \in \mathbb{R}^{2\times 3}}.$$

4.2.5 Local controller synthesis

Theorem 4.4 states that the controllability of system (4.1) with the gain matrix subject to a condition of type (4.37), (4.38), respectively, is equivalent with the controllability of a reduced order dynamical system. Specifically, in the case 1 of Theorem 4.4, the usual controllability tests (e.g. the gramian of controllability, controllability matrix) and design of gain matrix apply (e.g., pole placement or solving a Riccati equation). In the present paper we choose to construct the controller $\tilde{K}$ in (4.50) with a LQ design using the solution of the discrete algebraic Riccati equation

$$\tilde{A}^T P \tilde{A} - \tilde{A}^T P \tilde{B} (\tilde{B}^T P \tilde{B} + R)^{-1} \tilde{B}^T P \tilde{A} + Q = 0, \tag{4.53}$$

where we denote $\tilde{A} = A + B\Gamma$ and $\tilde{B} = B\Psi^T$. Furthermore, assuming the system (4.50) is controllable and a suitable gain matrix $\tilde{K} = R^{-1}B^T P$ is obtained, it is simple to introduce it in (4.48) and to obtain the stabilizing K for the original dynamics (4.1) which additionally provide certain invariance properties with respect to a given hyperplane.

In the case 2 of Theorem 4.4, several design approaches can be proposed. Here we propose a solution close to the ideas in [Crusius and Trofino, 1999], based on a LMI conditions.

Proposition 4.3. *Given the matrices $\tilde{\Gamma}$,B, $\tilde{\Psi}$, with B full column rank, any feasible solution in terms of matrices $G = G^T \succ 0$, M, N for*

$$\begin{cases} \begin{bmatrix} -G & \left(A + B\tilde{\Gamma}\right)^T G + \tilde{\Psi}^T N^T B^T \\ G\left(A + B\tilde{\Gamma}\right) + BN\tilde{\Psi} & -G \end{bmatrix} \prec 0, \\ BM = GB. \end{cases} \tag{4.54}$$

provides a gain matrix $\breve{K} = M^{-1}N\tilde{\Psi}$ stabilizing the system (4.46). □

Proof: If B is full column rank, then it follows from $BM = GB$ that M is also full rank, and thus invertible. This allows to obtain $B = GBM^{-1}$. Furthermore, the desire control law has the structure $u(k) = -M^{-1}N\tilde{\Psi}x(k)$. Exploiting these facts, from the first condition we obtain that

$$\begin{bmatrix} -G & ((A + B\tilde{\Gamma}) + B\breve{K})^T G \\ G((A + B\tilde{\Gamma}) + B\breve{K}) & -G \end{bmatrix} \prec 0, \tag{4.55}$$

or equivalently using Schur complement:

$$-G + ((A + B\tilde{\Gamma}) + B\breve{K})^T G((A + B\tilde{\Gamma}) + B\breve{K}) \prec 0,$$

which proves that system (4.46) is stabilizable via the proposed state feedback. □

Subsequently, the matrix $\breve{K}$ can be replaced in (4.51) to obtain the desired state feedback gain matrix K.

Illustrative example

Figure 4.6 resumes the theoretical details discussed in this section by a graphical illustration. Therefore, $S \in \mathbb{R}^2$ is the prohibited region defined as in (4.3). All the equilibrium states lie on the hyperplane which pass through the origin (see, (4.23) and Remark 4.6). Solving the optimization problem (4.41), we find an eigenvector which approximates the normal to one of the frontiers of the prohibited region. Moreover, the hyperplane partition the space into invariant half-spaces (see Lemma 4.2).

4.3 The global design problem

In this section we build on the results obtained in the previous sections. The goal is to define a control law which, given the system dynamics described by (4.1) and the prohibited region (4.3), transfers all the possible trajectories asymptotically to an equilibrium point lying *as close as possible* to the origin while respecting the constraints (4.6). With the results obtained in the previous sections, a local linear control feedback

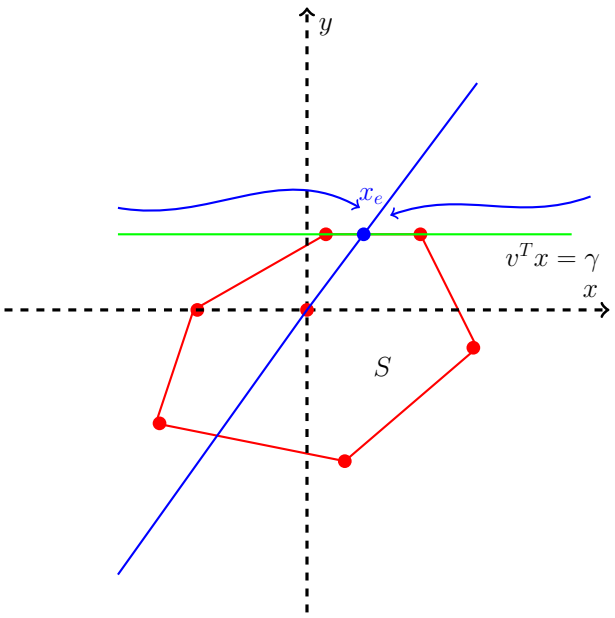


FIGURE 4.6: Prohibited region and equilibrium point lying on the boundary of the feasible region

gain is available such that x_e, a point on the frontier of S, is an attractor for the closed-loop unconstraint trajectories. In the constrained case, the condition $x(k) \notin S$ is assured only for a half-space, described by one of the supporting hyperplanes of S.

In the present section we will describe the procedure to ensure the stability of x_e by the use of a receding horizon optimal control procedure. Its design principles are related to the dual-mode control:

- a generic optimization-based control integrating collision avoidance constraints;
- its equivalence with the unconstrained feedback law (4.21) over an invariant region containing x_e;
- guarantees of convergence in finite time towards this invariant region.

Consider the system (4.1), the receding horizon the optimization problem to be solved is formulated as:

$$\begin{aligned}\mathbf{u}^* = &\arg\min_{\mathbf{u}}((x(k+N_p|k)-x_e)^T P(x(k+N_p|k)-x_e)+ \\ &+\sum_{s=1}^{N_p-1}(x(k+i|k)-x_e)^T Q(x(k+i|k)-x_e)+ \\ &+\sum_{s=0}^{N_p-1}(u^T(k+s|t)-Kx_e-u_e)R(u(k+s|k)-Kx_e-u_e)),\end{aligned} \tag{4.56}$$

$$\text{subject to : } \begin{cases} x(k+s+1|t) = Ax(k+s|k)+Bu(k+s|k), & s=0,\dots,N_p-1, \\ x(k+s|k)\in\mathbb{R}^n\setminus S, & s=1,\dots,N_p, \end{cases} \tag{4.57}$$

with $\mathbf{u}=\{u(k|k),\dots,u(k+N_p-1|k)\}$. The parameters x_e, u_e and K, are determined in the previous section (see (4.21)–(4.23), (4.37)–(4.50)). Applying the first component of the optimal formulation (4.56)–(4.57) and reiterating the optimization using the new state $x(k)$, considered measurable, we dispose of a global control law with the following properties (formulated here without the formal proofs which can be derived without difficulties based on the classical results in Proposition 4.2 and [Mayne et al., 2000] and [Chmielewski and Manousiouthakis, 1996]):

- the optimization problem is recursively feasible (as consequence of the unbounded feasible domain);
- it is tractable (finite number of constraints);
- the matrices P, Q, R can be tuned upon inverse optimality principles to ensure the equivalence between the unconstrained optimum and the feedback control action (4.21), $u(k)=K(x(k)-x_e)+u_e$;
- reachability analysis can be used to determine the minimal horizon N_p such that the predicted state trajectory

$$v^T x(k+N)\geq\gamma,\forall x(k)\in(R^n\setminus S).$$

In order to avoid the difficulties of the reachability analysis in the choice of the prediction horizon N_p, one can use the alternative receding horizon formulation which exploits a

"terminal set - terminal cost" stability argumentation:

$$\begin{aligned}\mathbf{u}^* = \underset{\mathbf{u}}{\arg\min}((&x(k+N_p|k)-x_e)^T P(x(k+N_p|k)-x_e)+ \\ &+\sum_{s=1}^{N_p-1}(x(k+s|s)-x_e)^T Q(x(k+s|s)-x_e)+ \\ &+\sum_{s=0}^{N_p-1}(u^T(k+s|k)-Kx_e-u_e)R(u(k+s|k)-Kx_e-u_e)),\end{aligned} \tag{4.58}$$

$$\text{subject to}: \begin{cases} x(k+s+1|k) = Ax(k+s|k)+Bu(k+s|k), & s=0,\dots,N_p-1, \\ x(k+s|k)\in\mathbb{R}^n\setminus S, & s=1,\dots,N_p, \\ x(k+N_p|k)\in v^T x(k|k)\leq\gamma. \end{cases} \tag{4.59}$$

Note that the last constraint ensures the invariance and the contractivity of the domain of attraction of the corresponding closed-loop feasible states for (4.59). Furthermore, in this formulation the adding of constraints on the input can be handled, the direct consequence being the reduction of the feasible domain.

The former construction depends explicitly upon cost matrices P, Q, R. Practically, the same matrices are used to provide (via the Riccati equation) the optimal gain K. Here however, we already proposed a value for the gain, based upon invariance assumptions. Thus, an inverse optimality reasoning becomes necessary. Having the control law $u(k) = Kx(k) + u_e$, we deduce (see, Larin [2003] for further details) a triplet of cost matrices (P, Q, R) for which the matrix K would be the solution of the corresponding Riccati equation. The inverse optimality problem can be addressed in a computationally attractive manner via the LMI formulation [Larin, 2003]:

$$\begin{cases} P\succeq 0, \\ A^TPA-P-K^TRK^T-K^TB^TPBK\preceq 0, \\ \begin{bmatrix} -Y & T \\ T^T & I \end{bmatrix}\succ 0, \quad T=RK+B^TPBK+B^TPA, \\ Y\prec\lambda I, \end{cases} \tag{4.60}$$

where Y is a symmetric matrix, λ is a scalar, I is a unit matrix of appropriate dimension. The LMI (4.60) provide a triplet (P, Q, R) as result of a feasibility problem. Note however that the cost matrices obtained are not uniquely defined.

On a more general note, a last aspect which need to be pointed out is that, for the same type of control problems a so-called "viability kernel" can be defined (the interested reader is referred to Definition 4.4.1, page 140, [Aubin et al., 2011]). Moreover, the viable capture basin of certain equilibrium will collect all the trajectories in a finite time while satisfying the constraints (see, for extensive details [Aubin et al., 2011]). In

practical terms, in these sets there exists a prediction horizon such that the trajectories can be guaranteed to reach a terminal region and thus assure the feasibility of the scheme.

4.4 Collision avoidance example

The following illustrative example will describe the limit behavior of an agent in the presence of adversary constraints. More precisely, the convergence to the relative position "zero" is impossible for the agent since a fixed convex obstacle contains the equilibrium position.

Let us consider a linear system, whose dynamics is described by:

$$A = \begin{bmatrix} -0.78 & 0.33 \\ -0.85 & 1.08 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 1 \\ -5 & 2 \end{bmatrix} \tag{4.61}$$

The components of the state are the position coordinates of the agent. Note that the pair (A, B) is stabilizable. The state constraints as described in (4.3) are illustrated in Figure 4.7 by the red polytope. Solving the optimization problem (4.41), we obtained an affine parametrization of the gain matrix

$$K = \begin{bmatrix} -0.17 & -0.09 \\ 0.74 & -0.38 \end{bmatrix}$$

as in (4.48) with

$$\tilde{K} = [-0.17 \ \ -0.09], \quad \Gamma = \begin{bmatrix} 0 & 0.86 \\ 0 & -0.31 \end{bmatrix} \quad \text{and} \quad \Psi = [1 \ 0.70].$$

This makes the closed-loop matrix to have a hyperplane of the prohibited region as an eigenvector. Furthermore, we obtained $u_e = [0.09 \ \ 1.29]$ and the equilibrium point $x_e = [0 \ \ 10.1]$, illustrated as a green dot in Figure 4.7. The tuning parameters of the optimization problem (4.56) are:

$$P = \begin{bmatrix} 0.59 & -0.04 \\ -0.04 & 0.50 \end{bmatrix}, \quad Q = \begin{bmatrix} 0.11 & 0.30 \\ 0.30 & 0.21 \end{bmatrix}, \quad R = \begin{bmatrix} 0.54 & -0.30 \\ -0.30 & 0.65 \end{bmatrix},$$

and the prediction horizon $N = 2$.

Finally, Figure 4.7 depicts three different state trajectories converging to a unique equilibrium point when the predictive control law (4.56) is applied.

Until now we have developed the conditions which force any trajectory of a linear dynamical system to remain outside of a given prohibited region. We have also discussed

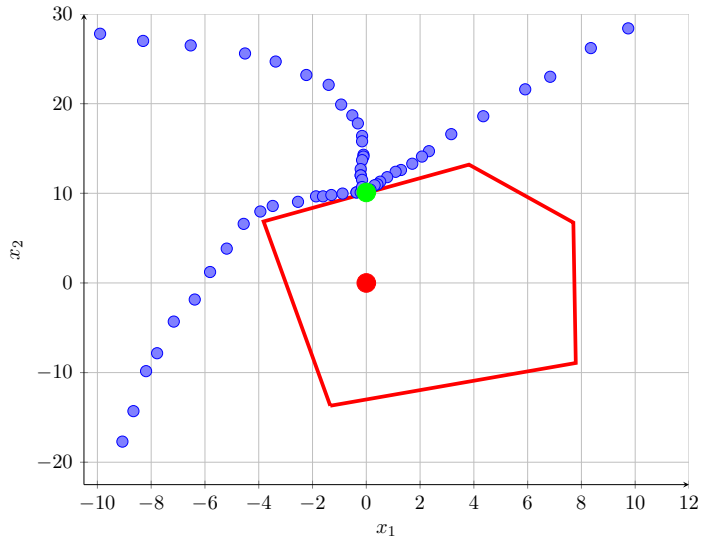


FIGURE 4.7: The prohibited region and different agent state trajectories which converge to a fixed point.

on the link between the conditions we have imposed and the PWA control law given by an MPC optimization problem.

4.5 Extension to multi-agent formation

Up to now we have discussed the limit behavior of an agent subject to geometric adversary constraints. Although the discussion was made for a LTI prediction model, it can be adapted to the multi-agent formulation. We speak here about the centralized formulation (see Section 3.4, Chapter 3) since this method allows a clear equivalence between the extended centralized system and the dynamics (4.1).

As explained in Section 3.3 the problem of finding a minimal configuration of agents in steady state is the result of a constrained optimization problem. The cost defines what we understand by "minimal configuration" and the constraints impose steady state behavior (each agent stays in a fixed position) and collision avoidance:

$$\min_{(x_i,u_i),\ i\in\mathcal{I}} \quad \sum_{i=1}^{N_a} \|x_i\|_2^2, \tag{4.62}$$

$$\text{subject to: } (x_i - x_j) \notin (\{-S_i\} \oplus S_j), \quad \forall i,j \in \mathcal{I},\ i \neq j, \tag{4.63}$$

with $\mathcal{I}$ denoting the collection of all agents indices. By applying a lifted state transformation (considering all the states into an "extended" state) $\mathbf{x} = \begin{bmatrix} x_1 & \dots & x_{N_a} \end{bmatrix}$ it

can be observed that (4.62)–(4.63) becomes similar to (4.5)–(4.6). There is a difference in the shape of the prohibited region. If for (4.5)–(4.6) the non-convex region was the complement of a convex set, here it becomes the complement of a union of convex sets: each collision avoidance constraint gives a different convex set which has to be avoided. Thus, we can conclude that formulations (4.5)–(4.6) and (4.62)–(4.63) are similar but they differ at the implementation level. We expect increased difficulties due to the extended state-space formulation and due to the presence of a union of convex sets. Then, solving the optimization problems (4.41)–(4.56) we can impose that the trajectories of the centralized system will converge towards a feasible and unique fixed point. Going back to the original space, we see that this fixed point from the extended space decomposes into fixed points for each agent. Thus, we have obtained a feasible and unique fixed formation. No matter the agents initial positions, they will converge into the same final configuration.

These results are guaranteed to hold for a centralized scheme where everybody communicates with everybody. Once we degrade the centralized form of the optimization problem, we start loosing these properties. In principle, and depending on the implementation, a distributed approach should still give convergence towards the formation in the sense that the duality gap should be zero. However, the constraints feasibility in the convergence stage is not guaranteed. We emphasize here also the importance of the computation of the robust positive invariant (RPI) safety region for an agent. This means that the bounded perturbation affecting an agent will not lead to collisions: the invariance assures that at any instant the real agent remains "close" to the nominal position.

4.5.1 Multi-agent formation example

As already mentioned, the method presented in the present chapter can also be extended to solve collision or obstacles avoidance problems for multiple agents. This implies at the modeling stage a compact representation of the obstacles and/or a safety region for an agent in terms of (4.3). Consequently, a safety region can be associated to each agent and imposes that the inter-agent dynamics do not overlap each individual restriction. It is important to assure that a control action will not lead to a cycling behavior which implies energy consumption. Formally, the fact that a set of agent remains in (or arrives at) a unique configuration, as a result of some suitable control strategy, is equivalent with saying that in an extended space, there exists and it is unique a point which can be made fixed through the same control strategy.

Without entering into an exhaustive presentation, we will make use of the techniques presented in the previous chapters, where the collision avoidance problem in the context of multi-agent formations is studied in detail. Consequently, here we will only illustrate that, by using the proposed method the agents converge to a unique configuration (i.e.,

to a unique equilibrium point in an extended state space). Figure 4.8 depicts the evolution of two heterogeneous agents with different associated safety regions (the blue and the red polytopes described as in (4.3)) and different initial positions. More precisely, as it can be seen in Figure 4.8, the objective of the formation is to reach the origin all by avoiding the superposition of the safety regions. The trajectories initiated in three pairs of initial conditions are presented. First scenario corresponds to a position $(-16, 35)$ for the red agent and $(-8, 35)$ for the blue agent. The equilibrium formation $\{x_e^1, x_e^2\} = \{(-1, 1); (1, 1)\}$ is achieved in about 60 steps with a MPC synthesized via a dual mode procedure with a prediction horizon $N_p = 2$. In the second scenario, the agents start from $(-28, -27)$, the red agent position and $(-21, -34)$, the blue agent position and they converge to the same equilibrium even if the maneuver is showing an aggressive contouring maneuver. Finally, the third scenario, where the red agent starts from $(24, -27)$ position and the blue agent from $(30, -26)$, confirms the result of the previously described cases.

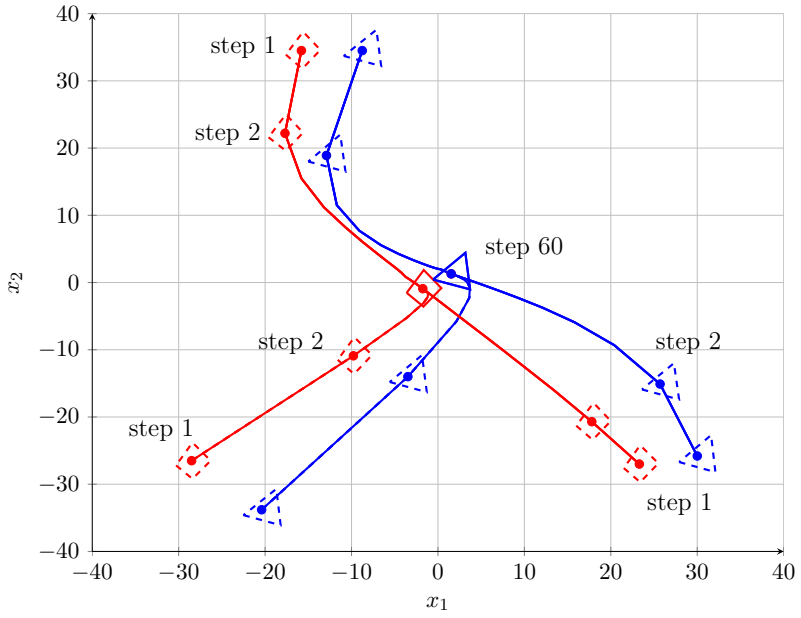


FIGURE 4.8: The evolution of the agents with different initial positions at three different time steps.

4.6 Concluding remarks

A finite horizon predictive optimization problem formulation was proposed in order to describe the evolution of a linear system in the presence of a set of adversary constraints.

This type of constraints are particular, as they make the convergence of the system trajectory to origin an infeasible task. We propose a dual-mode control law which switches between an unconstrained optimum controller and a local solution which handles the constraints activation, when necessary. Simple algebraic conditions for the existence and uniqueness of a stable fixed point on the boundary of the feasible region represent the main result of the present work, completed with an optimization based control for the global attractivity. The analyzed cases are presented through some illustrative examples and collision avoidance simulation results. Furthermore, the analysis was extended for multi-agent formation. In this context, the previously described fixed point uniqueness condition translate in the multi-agent formulation into a unique configuration condition.

Chapter 5

Examples, simulations, benchmarks and applications

The present chapter looks towards a particular class of applications popular in the multi-agent framework. The first part of the chapter is dedicated to the validation of the predictive control design for the flight control of Unmanned Aerial Vehicles (UAVs). The second part extends via simulations the multi-agent dynamical systems study.

5.1 Flight control experiments of Unmanned Aerial Vehicles

Unmanned Aerial Vehicles (UAVs) are flying devices that have received increasing attention in recent years, particularly after the tragic events of September 11, 2001, when the world has questioned the safety of the onboard aircrafts [Valavanis, 2007]. Currently, most UAVs are used for military mission planning for unmanned combat support, traffic monitoring, surveillance and search for survivors. Also, the range of possible civilian applications for UAVs is expending. Many of the enabling technologies developed for military UAVs are becoming similar or identical to those required for civil UAVs.

One of the main research challenges is to improve and increase their autonomy, in particular to empower the application of advanced control design methods. Their aim is to respect constraints of the vehicle dynamics and to permit the reconfiguration of the vehicle trajectory in case that unexpected events occur in the system. The combined use of Model Predictive Control (MPC) with flatness concepts represents a challenging combination in the state-of-the-art allowing to handle the real-time control, the trajectory

generation and can cope with the robustness issues using set-theoretic methods. Moreover, to the best of the author' knowledge, such a strategy has not been evaluated in flight tests on actual vehicles. We will emphasize in the rest of the chapter the practical implementation of these methods for a real UAV system.

There exist various techniques for the real-time control of UAVs. Usually low-complexity (and robust) control methods are preferred such as, sliding mode control [Bencatel et al., 2011], dynamic programming approach for motion control of autonomous vehicles [da Silva et al., 2007] or control strategies that first estimate the speed and heading of the vehicles and then use simple feedback control laws for stabilizing different measurement errors [Soares et al., 2012]. In comparison to these solutions we will show that the presented MPC approach, meet the real-time computational constraints even if it presents a relatively important real-time computational load.

There are also various applications in the literature where real-time MPC is applied to vehicle maneuvering problems. For example, [Keviczky and Balas, 2006] uses a predictive guidance controller for an autonomous UAV and a fault detection filter for taking into account the disturbances. Mixed-Integer Programming (MIP) techniques combined with receding horizon strategy was useful to coordinating the efficient interaction of multiple UAVs in scenarios with many sequential tasks and tight timing constraints (see, [How et al., 2004], [Schouwenaars et al., 2005]). Furthermore, some works investigate the capability of Nonlinear MPC for tracking control. Among these contributions, [Kim et al., 2002] formulates a nonlinear MPC algorithm combined with the gradient-descent method for trajectory tracking, and [Fontes et al., 2009] proposes a two-layer control scheme composed by a nonlinear and a linear predictive controller for a group of nonholonomic vehicles moving in formation. However, it is important to point out that nonlinear MPC is computationally involved and thus, not necessarily suitable for a wide class of applications. This highlights the importance of developing simpler real-time optimization problems embedded within predictive control formulations for plants described by nonlinear models. In this sense, the authors of [Falcone et al., 2007] consider a MPC tracking controller based on successive *on-line* linearizations of the nonlinear model of the corresponding plant. Yet, the computational complexity of the proposed MPC scheme remains significant. The approach advocated in the present work follows a similar linearization principle from the prediction model point of view, but avoids its real-time computation by the use of a precomputed Voronoi diagram of the linearized models. This novelty is used in conjunction with an efficient trajectory generation mechanism in order to reduce the real-time computations.

The structure of flatness plays an important role in the control of such systems, with practical design algorithms for motion planning, trajectory generation, and stabilization [Rouchon et al., 2003]. Among the applications, [Hao and Agrawal, 2005] propose a combination between differential flatness and a graph search method for the on-line planning and control of multiple ground mobile robots with trailers moving in groups. Also, the authors in [Van Nieuwstadt and Murray, 1998] apply a real-time trajectory

FIGURE 5.1: "Cularis" UAV of LSTS Department, University of Porto.

generation algorithm based on flatness and receding horizon without constraints to a thrust vectored flight experiment (the Caltech ducted fan). Finally, the authors in [De Doná et al., 2009] develop a receding horizon-based trajectory generator methodology for generating a reference trajectory parameterized by splines, with the property that it satisfies performance objectives.

Even if our design choice is based on the manipulation of flat outputs, is worth to be mentioned that, for the cases where flat outputs are not available, alternative trajectory generation methods can be used. We mention, for example a computational approach to generate real-time optimal trajectories by using a NonLinear Trajectory Generation software package [Milam et al., 2002] or the ACADO toolbox which permits to compute a reference trajectory as the solution of an optimization problem with constraints [Houska et al., 2011].

In the current chapter we present software-in-the-loop simulations and real-time flight tests results for the predictive control of Unmanned Aerial Vehicles (UAVs). The flight experiments took place at the Portuguese Air Force Base OTA in May 2012 on small platforms (Figure 5.1, Figure 5.2, Figure 5.3) owned by the Air Force Academy and LSTS laboratory, University of Porto.

In the following, a summary of the control setup used onboard the UAV is presented. The testbed implements a ground control, that is, the controller runs on Matlab on a laptop that receives telemetry and sends flight commands through a Ground Station. The autopilot system runs Piccolo II (see, [Vaglienti et al., 2011]), which sends and receives airspeed and bank commands from DUNE, a control software developed by the LSTS lab from University of Porto [Pinto et al., 2012].

The LSTS lab testbed software and hardware architectures enabled us to integrate and to test our predictive control algorithms in a real-time environment. One fundamental control issue we deal with in this chapter is the problem of tracking a given reference

FIGURE 5.2: "Alfa 06" UAV of Portuguese Air Force Academy, OTA.

FIGURE 5.3: "Pilatos 3" UAV of LSTS Department, University of Porto.

trajectory in the presence of constraints (see, [Valavanis, 2007], [Aguiar and Hespanha, 2007] and the references therein). This problem is even more challenging because most of the UAVs dynamical systems are nonlinear, underactuated and exhibit nonholonomic constraints [Li and Canny, 1993], [Reyhanoglu et al., 1999].

We concentrate next to the description of the control system design, the three-degree-of-freedom (3 DOF) aircraft model used to test the control system, and the implementation of the controller intended to run in real-time in an autonomous procedure. A specified trajectory is generated for a vehicle using the *differential flatness formalism*. The proposed trajectory generation mechanism takes into account way-point conditions and furthermore, allows us to obtain *off-line* linearizations of the nonlinear vehicle model along the flat trajectory. Since the reference trajectory is available beforehand, a *real-time optimization problem* which *minimizes* the *tracking error* for the vehicle is solved based on a prediction of the future evolution of the system, following the model-based control principles.

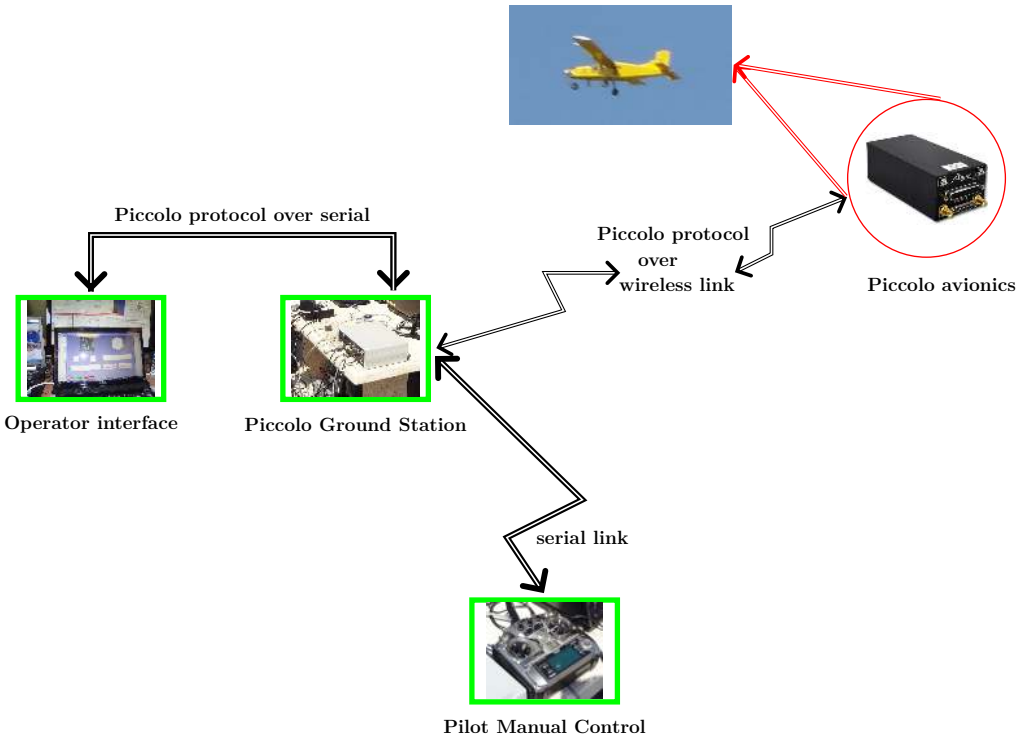


FIGURE 5.4: Operational control system setup of the UAV.

The validation procedure of the proposed method will test the performances by software-in-the-loop simulations and subsequently by the ground test results for the control of an autonomous Unmanned Aerial Vehicle (UAV). Finally, in the subsequent sections we address all the issues concerned with the delays in communication and/or the connectivity between the Ground Station and the autonomous flight control computer.

5.1.1 Testbed hardware and software architecture

The UAV platform is controlled by a highly integrated, user customizable Piccolo system which is manufactured by *Cloud Cap Technologies* (`http://www.cloudcaptech.com/piccolo_system.shtm`). The Piccolo control system setup, shown in Figure 5.4, consists of four main parts [Vaglienti et al., 2011]:

- an Avionics control system located onboard the UAV;
- a Ground Station;
- a Pilot Manual Control;

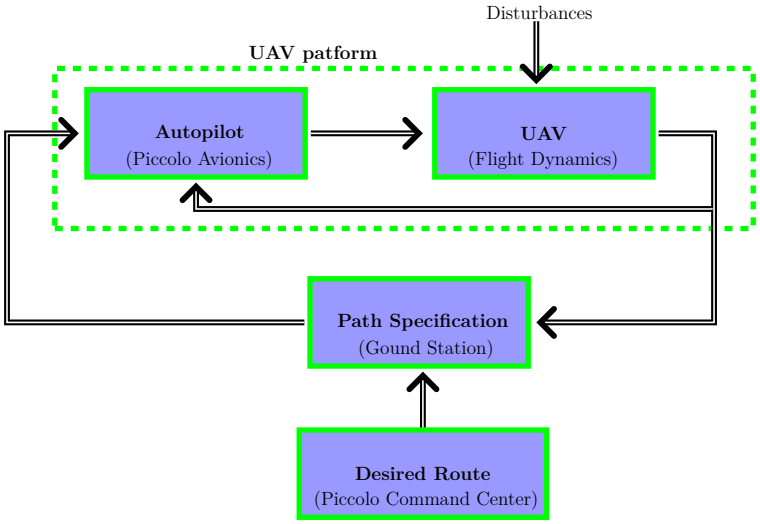


FIGURE 5.5: Piccolo closed loop control system.

- the Piccolo Command Center (operator interface).

These four elements provide a reliable way to fly the UAV and enable the end user to program desired routes for the UAV via way-points.

As shown in Figure 5.5, the Piccolo autopilot setup has two control loops, that is, a faster inner loop on board the UAV and a slower outer loop, which is implemented on the Ground Station. The outer loop provides the path to be followed by the vehicle whereas, the inner loop controls the dynamics of the UAV. Furthermore, Piccolo autopilot relies on a mathematical model parameterized by the aircraft geometric data and has a built-in wind estimator. A slightly more detailed description of the main components of the Piccolo autopilot is given in the following.

5.1.1.1 Piccolo autopilot description

The **Piccolo Avionics** system is an autopilot designed to track the controlled path transmitted by the Ground Station. It relies on a group of sensors which includes [Almeida et al., 2007]: three rate gyroscopes and two axis accelerometers, a radio, a Global Positioning System (GPS) to determine its geodetic position, and a set of dynamic and static pressure sensors coupled with a thermometer to determine the airplane's true airspeed and altitude.

Besides the GPS receiver the autopilot includes also an IMU (Inertial Measurement Unit). The GPS delivers X, Y, Z position and velocity measurements, with relatively

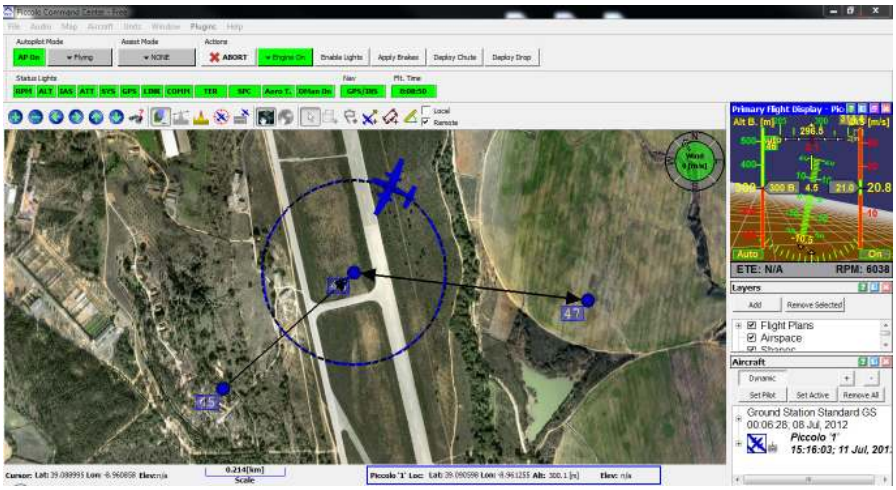


FIGURE 5.6: Piccolo Command Center showing flight plan and actual position of the UAV.

larger errors in the positions and relatively small errors in the velocities. The IMU delivers precise position measurements [Almeida et al., 2007].

A Kalman filter approach is used to gather the information provided by the IMU block (accelerations and angle-rates) and the GPS block (positions and velocities) in order to estimate the UAV full state. Both blocks are taken into consideration because the IMU delivers the best inputs, but requires the GPS support to compensate for the angle estimation drift. Furthermore, an embedded Power PC receives the state information from all sensors and runs autopilot loops commanding the control surfaces actuators of the airplane (ailerons, elevator and rudder), the engine's thrust as well as playload ports. Telemetry data is transmitted to the Ground Station through a radio modem sending the information at a rate of 25 Hz (conversely, this means on average, a discretization step of $1/25 = 0.04$ seconds). With omni-directional antennas the signal strength is enough for a 3 km communication radio. On the other hand, with directional antennas, the communication range extends to 40 km.

The **Piccolo Ground Station** has two very important roles in the system. Firstly, it provides the communication link between the Piccolo Command Center, the Pilot Manual Control and the Piccolo Avionics. Secondly, converts the intentions of the end user captured through the operator interface, into meaningful commands for the autopilot. The ground station can concurrently monitor up to 10 UAVs. Moreover, it performs differential GPS corrections, and updates the flight plan, which is a sequence of three dimensional way-points connected by straight lines.

The **Piccolo Command Center** (operator interface) consists of a portable computer and a custom developed software which allows the end user to configure and to operate

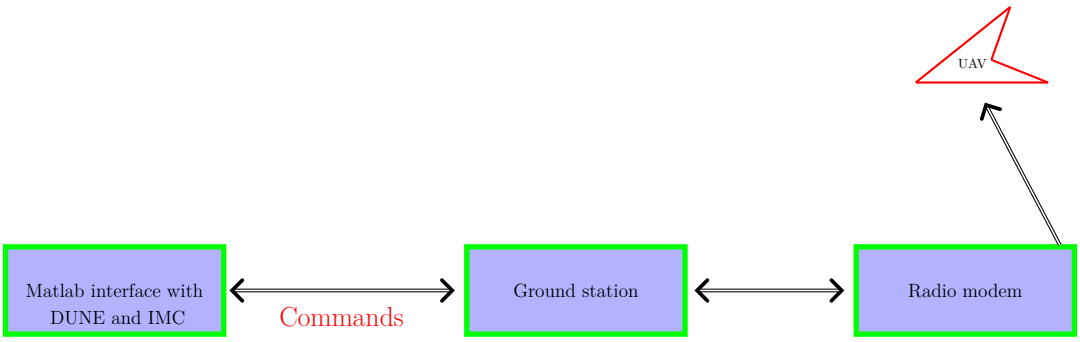


FIGURE 5.7: Software architecture.

the Piccolo System. Furthermore, the graphical user interface (see, Figure 5.6) developed by Could Cap shows flight information and allows the operator to monitor the flight progress, to program a desired route for the UAV via way-points, to send velocity, bank or altitude references and finally, to configure the desired gains of the control system in the autopilot.

The **Pilot Manual Control**, which is mainly used for take-off and landing, provides the end user with a way to override the commands generated by the Ground Station and allows a qualified UAV Pilot to take control of the UAV.

Finally, Piccolo environment can be understood as the "firmware" with a collection of protocols which enable the use of the UAVs. In the forthcoming section we present shortly the upper level management represented by the software architecture that served as an integration framework for flight code development and simulations in a real-time environment. Once these elements will be presented, we will be able to concentrate explicitly on the control design, which is the main goal of the present chapter.

5.1.1.2 Software architecture

The UAV operational system block diagram, including the software architecture is depicted in Figure 5.7.

LSTS lab interfaced Piccolo software with MATLAB using DUNE and IMC messaging system [Pinto et al., 2012]. DUNE (Unified Navigational Environment) it is a generic embedded software used to compile code for vehicle control, navigation, communication, sensor and actuator access. Some examples of software tasks that can be implemented by DUNE are for instance:

- providing a network gateway for the connection to CloudCap software;

- handling the physical/simulation connection to the autopilot or the Ground Station;
- decoding and encodes telemetry and control packets from Piccolo and other onboard systems (e.g., gimbal camera), from and to Inter-Module Communication (IMC) packets [Martins et al., 2009].

DUNE can run in simulation mode, which disables all the sensor and actuator drivers and replaces them with simulating tasks. It may also run in hardware-in-the-loop mode, which allows for some sensor or actuator drivers to be enabled, together with simulating tasks [Pinto et al., 2012].

The Inter-Module Communication (IMC) is a message-oriented protocol designed and implemented for communication among heterogeneous vehicles (for example, aerial and marine vehicles), sensors and human operators [Gonçalves et al., 2011].

Is noteworthy that the core control algorithm can run in Matlab. Both, the input to the control algorithm and the output of the control algorithm are communicated in terms of IMC messages. The DUNE interprets the IMC messages to the corresponding commands in Piccolo software which is communicated to the UAV [Martins et al., 2009], [Dias et al., 2010].

5.1.2 UAV model in view of control design

The autopilot in the Piccolo avionics is responsible for the low level control, stabilization, as well as the flight plan navigation and tracking of the way-points. It consists of seven PID loops and a turn compensator. A short description is presented in Figure 5.8, for more details the reader is referred to [Vaglienti and Niculescu, 2004]. The inner loops control a series of relevant quantities as, airspeed, altitude, turn rate. The quality of the low level inner control loops allows to abstract it away when the global dynamics are to be modeled.

The airplane model is a nonholonomic system. It is completely controllable (if the velocity is different than zero), but it cannot make instantaneous turns in certain directions. This means that the vehicle state depends on the path executed until the current moment. Further, the feasible path depends on the aircraft's current state, as forces cannot be imposed in any direction concomitantly. The two main control forces of the aircraft are the lift and the thrust. Thrust magnitude is controlled directly through the motor power. Lift magnitude is controlled indirectly through the aircraft angle of attack and the aircraft airspeed. The framework presented here can be adapted to vehicles moving in 2D or 3D. Subsequently, the airplane can be represented by the following simplified kinematic models[1] with different Degrees-of-Freedom (DOF) [Bencatel et al., 2011]:

[1] In the present work, we adopt a kinematic model for the airplane that is not a simple double integrator. Instead, the model is written in terms of the vehicle's speed, heading and the bank.

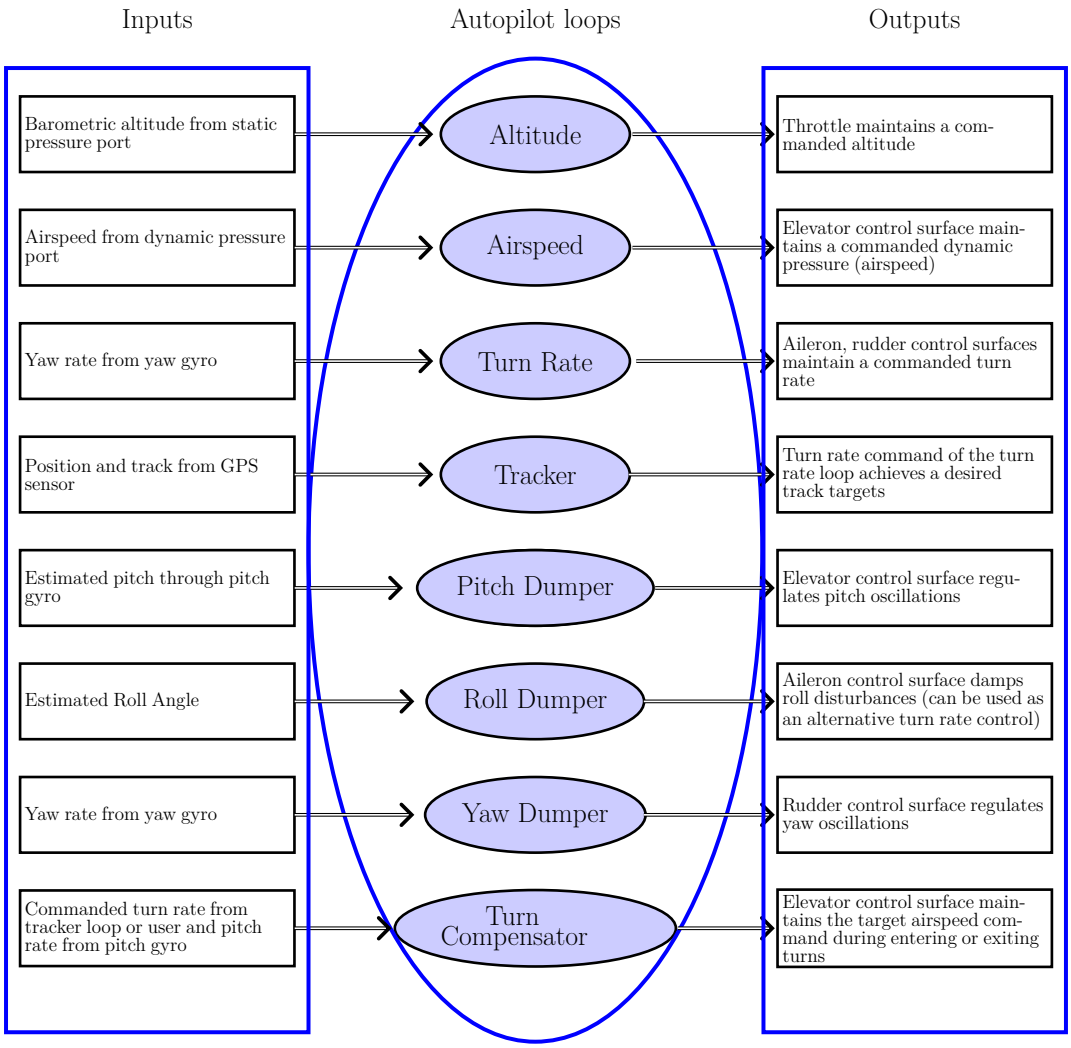


FIGURE 5.8: Piccolo autopilot loops.

2D model with 3-DOF

$$\begin{aligned} \dot{x}(t) &= V_a(t)\cos\Psi(t) + W_x, \\ \dot{y}(t) &= V_a(t)\sin\Psi(t) + W_y, \\ \dot{\Psi}(t) &= \frac{g\ \tan\Phi(t)}{V_a(t)}, \end{aligned} \tag{5.1}$$

3D model with 4-DOF

$$\begin{aligned} \dot{x}(t) &= V_a(t)\cos\Psi(t) + W_x, \\ \dot{y}(t) &= V_a(t)\sin\Psi(t) + W_y, \\ \dot{\Psi}(t) &= \frac{g\ \tan\Phi(t)}{V_a(t)}, \\ \dot{h}(t) &= \dot{h}_c(t). \end{aligned} \tag{5.2}$$

In the present work we explore the case of a 2D 3-DOF model (5.1) of an airplane in which the autopilot forces coordinated turns (zero side-slip) at a fixed altitude. The state variables are represented by the position $(x(t), y(t))$ and the heading (yaw) angle $\Psi(t) \in [0, 2\pi]$ rad. The input signals are the air-relative velocity $V_a(t)$ and the bank (roll) angle $\Phi(t)$, respectively. Also, the airspeed and the bank angle are regarded as the autopilot pseudo-controls. Furthermore, we assume a nearly null angle of attack and that the autopilot provides a higher bandwidth regulator for the bank angle, making its dynamics negligible when compared to the heading dynamics. W_x and W_y are the wind velocity components on the x and y axis. Notice that the 3D 4-DOF model is the same as the 2D 3-DOF model of the airplane, except for the altitude variation $h(t)$. In this case, besides the airspeed and the bank angle, the vertical rate $\dot{h}_c(t)$ is regarded also as the autopilot pseudo-control.

Our main objective being the design of a predictive control strategy, a reference trajectory needs to be available beforehand at least for a finite prediction window. Therefore, in the following, we use flatness concepts [Fliess et al., 1995] in order to provide flat states and inputs of the nonlinear system (5.1) at the pre-design stage (trajectory generation), which can be updated in real-time in order to allow rescheduling and target moves.

5.1.3 Flat trajectory generation

Consider notations:

$$\xi(t) = \begin{bmatrix} x^T(t) & y^T(t) & \Psi^T(t) \end{bmatrix}^T, \tag{5.3}$$

$$u(t) = \begin{bmatrix} V_a^T(t) & \Phi^T(t) \end{bmatrix}^T, \tag{5.4}$$

denoting the state vector and the input vector, respectively. Then, the general system (5.1) can be described as:

$$\dot{\xi}(t) = f(\xi(t), u(t)), \tag{5.5}$$

where $f(\cdot, \cdot) : \mathbb{R}^3 \times \mathbb{R}^2 \to \mathbb{R}^3$ is the state vector field.

In the following, we require the determination of a reference trajectory $(\xi^{ref}(t), u^{ref}(t))$ that steers the model (5.1) from an initial state $\xi^{ref}(t_0)$ to a final state $\xi^{ref}(t_f)$, over a fixed time interval $[t_0, t_f]$. Since the aforementioned system is controllable and allows the existence of flat outputs (see, for instance [Van Nieuwstadt and Murray, 1998], [De Doná et al., 2009]), we consider flatness properties in order to construct the required reference trajectory.

The systems' state and input will be represented as functions of a finite dimensional mapping $z(t)$ and a finite number of its derivatives (in this particular case it will be

shown that the second order derivative suffices):

$$\begin{aligned} \xi^{ref}(t) &= \eta_0(z(t), \dot{z}(t)), \\ u^{ref}(t) &= \eta_1(z(t), \dot{z}(t), \ddot{z}(t)). \end{aligned} \tag{5.6}$$

In the case of the dynamics (5.1), the vector $z(t) = \begin{bmatrix} z_1(t) & z_2(t) \end{bmatrix}^T \in \mathbb{R}^2$, called the flat output is defined as:

$$\begin{aligned} z_1(t) &= x(t), \\ z_2(t) &= y(t). \end{aligned} \tag{5.7}$$

It can be shown that, the corresponding reference state and input for the system (5.5) are obtained by replacing the reference flat output (5.7) into equations (5.6):

$$\xi^{ref}(t) = \begin{bmatrix} z_1(t) & z_2(t) & \arctan\left(\frac{\dot{z}_2(t)}{\dot{z}_1(t)}\right) \end{bmatrix}^T, \tag{5.8}$$

$$u^{ref}(t) = \begin{bmatrix} \sqrt{\dot{z}_1^2(t) + \dot{z}_2^2(t)} & \arctan\left(\frac{1}{g}\frac{\ddot{z}_2(t)\dot{z}_1(t) - \dot{z}_2(t)\ddot{z}_1(t)}{\sqrt{\dot{z}_1^2(t)+\dot{z}_2^2(t)}}\right) \end{bmatrix}^T, \tag{5.9}$$

where $t \in [t_0, t_f]$.

For a practical implementation, the flat output signal $z(t)$ is seen as a weighed sum of functions in a predefined basis. Imposing boundary constraints (or more generic way-points) for the evolution of the differentially flat systems (see, for instance [De Doná et al., 2009]) a flat output $z(t)$ can be generated by the resolution of a *linear* system of equalities. Exploiting these principles, our approach is to further introduce a set of way-points through which the vehicle must pass[2] in the interval $[t_0, t_f]$:

$$\mathbb{P} \triangleq \{p^i = (\xi^i, u^i), \quad i = 0, \ldots, N_w\}, \tag{5.10}$$

where N_w is the number of chosen way-points.

The list of way-points is assumed as being provided by an operator (which can oversee the operation) and incorporates control requirements: obstacle avoidance, check points, etc. Note that producing way-points in the reference trajectory generation is coherent with the existing software-hardware configuration which used way-points in the communication protocol (see previous description of Piccolo hardware and software architecture).

It is important to point out that polynomial basis functions are a poor choice because their dimension (degree) depends on the number of constraints imposed upon the inputs, states and their derivatives. This means that they are sensitive to the number of way-points, which can lead to an increased numerical sensitivity when t_f grows. More precisely, in this case the trajectory needs to be computed on segments (i.e., each segment

[2]Hereafter whenever we use the subscript we refer to time and when we use the superscript we index a point from a set of points.

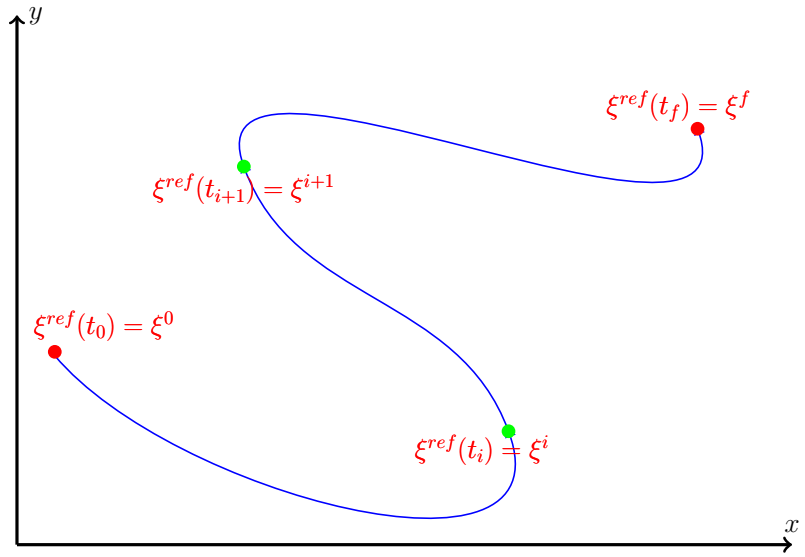


FIGURE 5.9: Flat trajectory which passes through 4 way-points.

taken between two consecutive way-points). Therefore, additional equality constraints are imposed on the flat output, leading to an increased number of polynomial basis functions beyond reasonable computation limits. For example, in our case, we need to go to higher order monomials in order to guarantee that there exists a weighted combination which satisfies all the equality constraints.

To overcome these issues, we used B-splines functions [De Doná et al., 2009], [Suryawan et al., 2010] which are (arguably) one of the best choices in the sense that their degree does not depend on the number of way-points. Actually, the degree depends only up to which derivative we want to assure continuity, this being in contrast with the polynomial basis. In our particular case, the third degree order of the B-splines suffices to assure smooth bank and velocity control inputs. Third degree order suffices because the bank control is defined by the first and second derivatives (see equation (5.9)), which means that the bank control is smooth only if second and third order derivatives of the flat output are continuous. As an illustrative example Figure 5.9 shows a flat trajectory which passes through 4 way-points as in (5.10).

Once we have at our disposal a reference trajectory we can now concentrate on the real-time aspects. Therefore, in the predictive control context we need to be able to handle the discretized and linearized model of the vehicle along a reference trajectory. In the following, we describe the proposed linearization strategy of the nonlinear system (5.1).

5.1.4 Linearization of the UAV model

For computation purposes, it is convenient to use the discretized model of the nonlinear system (5.5):

$$\xi(k+1) = f^d(\xi(k), u(k)). \tag{5.11}$$

For the time discretization we used the Euler explicit method, where we calculate the state of the system at a later time from the state of the system at the current time:

$$\xi(k+1) = \xi(k) + h \cdot f(\xi(t), u(t))|_{t=k\cdot h}, \tag{5.12}$$

where h is the discretization step. Even if very simple, the Euler method proved to be adequate. Of course, one can think of high order discretization schemes and, in particular, it would be interesting to have a discretization which is adapted to a variable discretization step (as it is the case due to the communication delays we experienced in the practical experiments).

Remark 5.1. Since the communication frequency of the Piccolo module was at most 25 Hz it follows that, we were able to use a step size as low as 0.04 seconds. In practice and because the maximum communication frequency is not always attainable (for more details see Section 5.1.6) we have chosen a value of 0.1 seconds for the length of the discretization step. □

In the sequel, we consider the linearization problem of the nonlinear discretized system (5.11). We take here a piece-wise affine (PWA) approach[3], that is, we consider a collection of points along the reference trajectory in which we pre-compute linear approximations of (5.11):

$$\mathbb{L} \triangleq \{l^j = (\xi^j, u^j), \quad j = 0, \ldots, N_l\}, \tag{5.13}$$

with N_l the number of chosen linearization points.

For a given point $l^j \in \mathbb{L}$ we consider the following Taylor decomposition:

$$f^d(\xi(k), u(k)) = f^d(\xi^j, u^j) + A_j(\xi(k) - \xi^j) + B_j(u(k) - u^j) + \beta_j(\xi(k), u(k)), \tag{5.14}$$

where the matrices $A_j \in \mathbb{R}^{3\times 3}$ and $B_j \in \mathbb{R}^{3\times 2}$ are defined as

$$A_j = \frac{\partial f^d}{\partial \xi}|_{\xi^j, u^j}, \qquad B_j = \frac{\partial f^d}{\partial u}|_{\xi^j, u^j} \tag{5.15}$$

[3]The PWA approach has received and extensive interest in the literature, representing a powerful technique for approximating the nonlinear systems and furthermore, proving their equivalence to other classes of hybrid systems. For a wide extent on the subject, the interested reader is referred to the work of [Sontag, 1981], [Heemels et al., 2001], [Ulbig et al., 2010] and the references therein.

and $\beta_j(\xi_k, u_k) \in \mathbb{R}^3$ represents the terms of the Taylor decomposition of rank greater than 1 (i.e., the nonlinear residue of the linearization):

$$\beta_j(\xi(k), u(k)) = f^d(\xi(k), u(k)) - f^d(\xi^j, u^j) - A_j(\xi(k) - \xi^j) - B_j(u(k) - u^j), \quad (5.16)$$

for all $j = 0, \ldots, N_l$. Therefore, the system (5.11) can be linearized in $l^j \in \mathbb{L}$ by the following dynamics:

$$\xi(k+1) = f_j^d(\xi(k), u(k)) \triangleq A_j\xi(k) + B_j u(k) + r_j, \quad (5.17)$$

with the affine constant terms $r_j \in \mathbb{R}^3$ defined as:

$$r_j = f^d(\xi^j, u^j) - A_j\xi^j - B_j u^j, \quad (5.18)$$

for all $j = 0, \ldots, N_l$.

In the following we consider a procedure of selecting between the predefined linearization points (5.13) for the current input/state values. To this end, we partition the state-space into a collection of *Voronoi cells*:

$$\mathcal{V}_j = \left\{\xi : ||\xi - \xi^j|| \leq ||\xi - \xi^r||, \ \forall r \neq j\right\}, \quad (5.19)$$

where each cell consists of all points whose linearization error is lower with respect to linearization around point ξ^j than with respect to any other point ξ^r from $\mathbb{L}$, with $r, j = 0, \ldots, N_l$. This allows a practical criterion for the selection of the linearization point during runtime:

$$\text{if} \quad (\xi, u) \in V_j \quad \text{then} \quad \xi(k+1) = f_j^d(\xi(k), u(k)), \quad \forall (\xi(k), u(k)) \in \mathcal{V}_j. \quad (5.20)$$

Is worth mentioning that the Voronoi decomposition is unique (by its geometrical properties) and, as such, it offers a generic design tool for any disposition of the linearization points. The drawback is that this criterion is purely geometric and do not take into account the dynamical properties of the model. This disadvantage can be mitigated by two practical procedures: the increase of the number of linearization points and the computation of the maximal linearization error (see [Fagiano et al., 2009] for a discussion on the accuracy of the linearization and the correspondence with a stabilizing control law). Since $\beta_j(\xi(k), u(k)) = f^d(\xi, u) - f_j^d(\xi, u)$ it follows that the linearization error is related to the topology of its corresponding cell, $\mathcal{V}_j$:

$$||\beta_j(\xi(k), u(k))|| \leq \max_{(\xi,u)\in\mathcal{V}_j} ||f^d(\xi, u) - f_j^d(\xi, u)||. \quad (5.21)$$

Basically, a Voronoi decomposition with decreasing volume of the cells leads to an increasing quality of the PWA approximation for the function (5.11).

The following remarks are in order.

Remark 5.2. An a priori computation of the linearization (5.15), (5.16) and (5.18) in all feasible combinations of inputs and states is difficult to handle. As such, we prefer to select the linearization points (5.13) along the flat trajectory under the assumption (to be verified along the system functioning) that the real trajectory will stay in the corresponding validity domain (Voronoi cell) and thus, the chosen linearization points will remain relevant to the problem at hand. □

Remark 5.3. Here we have chosen the linearization points equidistantly along the reference trajectory. This choice is acceptable as long as the trajectory tracking error is contained by similar uncertainty bounds over the associated Vornoi cells. Adaptive curve sampling can be employed via different parametrization in order to select these points. For example, the selection of linearization points can be seen as an optimization problem where the goal is to position the points in such a way as to minimize the linearization errors $\beta_j(\xi(k), u(k))$ in (5.14). Such an approach becomes relevant when the control problem is specified in a high dimensional space and the automatic treatment needs to be automatic. □

Proof of concept: In order to better explain the linearization strategy, we illustrated in Figure 5.10 the continuous trajectory of the nonlinear system (5.5) (in blue) and the piecewise linearized trajectory (in red). Therefore, the linear system (5.17) describes dynamics of the deviations of the real nonlinear system trajectories (5.11) from the reference state trajectory $\xi^{ref}(t)$ described by (5.8) at the application of an input reference signal $u^{ref}(t)$ in the from (5.9). Several linearization points (denoted as black dots)[4] have been considered for the construction of the Voronoi cells according to relationship (5.19). □

By linearizing the reference trajectory one has available of the necessary modeling framework ((2.9), Section 2.2, Chapter 2) for the feedback control part of the trajectory tracking according to the MPC principles ((2.1)–(2.2), Section 2.1, Chapter 2). This problem becomes the central objective for the remaining part of the chapter and will be detailed in the forthcoming sections.

5.1.5 Trajectory tracking control problem

Since the reference trajectory is available beforehand (through the use of flatness procedures), an optimization problem, which includes the minimization[5] of the vehicle tracking error, can be formulated in a predictive control framework. Practically, the vehicle will be controlled in real-time to follow the reference trajectory using the available information over a finite time horizon in the presence of constraints.

[4]Note that the way-points (5.10) can also be considered between the linearization points (5.13). Moreover, the linearization points and the way-points in the trajectory generation need not to be correlated.

[5]The nominal trajectory is conceived to respect state and input constraints, but the real vehicle state may not follow exactly the reference trajectory, although it is desirable to remain as close as possible to it.

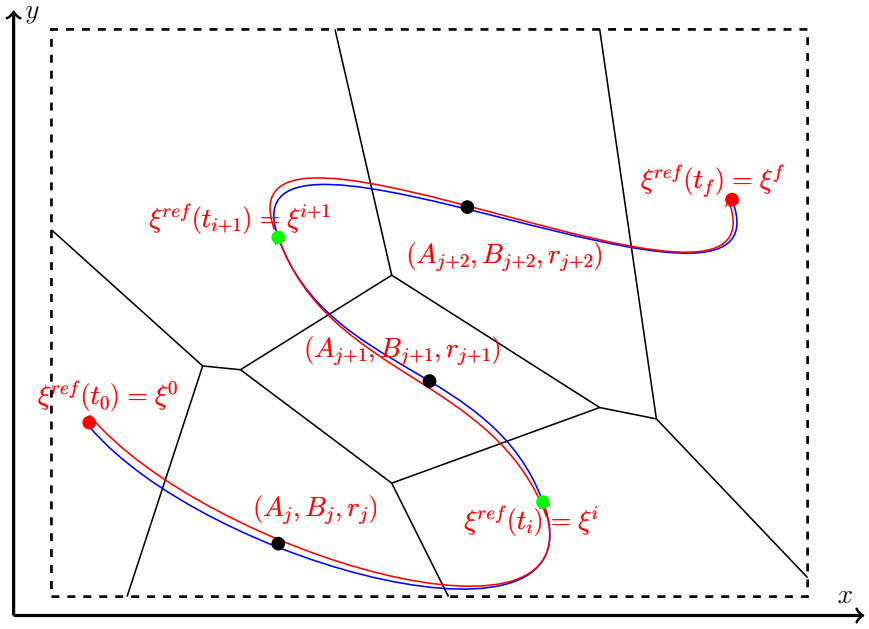


FIGURE 5.10: Real and linearized trajectories and the bounded Voronoi cells.

For the implementation we consider the recursive construction of an optimal open-loop control sequence $\mathbf{u} = \{u(k|k), u(k+1|k), \cdots, u(k+N_p-1|t)\}$ over a *finite* constrained receding horizon, which leads to a feedback control policy by the effective application of the first control action as system input:

$$u^* = \arg\min_{\mathbf{u}} \sum_{s=0}^{N_p-1} (||\xi(k+s|k) - \xi^{ref}(k+s|k)||_Q + ||u(k+s|k) - u^{ref}(k+s|k)||_R + \\ + ||\Delta u(k+s|k)||_{R_\Delta}), \tag{5.22}$$

subject to the set of constraints:

$$\begin{cases} \xi(k+s+1|k) = A_j\xi(k+s|k) + B_j u(k+s|k) + r_j, \\ \Delta u(k+s|k) = u(k+s|k) - u(k+s-1|k), \\ \xi(k+s|k) \in \mathcal{X}, \ s = 1, \ldots, N_p - 1, \\ u(k+s|k) \in \mathcal{U}, \ s = 1, \ldots, N_p - 1, \\ \Delta u(k+s|k) \in \mathcal{U}_\Delta, \ s = 1, \ldots, N_p - 1, \end{cases} \tag{5.23}$$

with the index j selected such that $(\xi(k+s|k), u(k+s|k)) \in \mathcal{V}_j$ as defined in (5.19), for some $j \in \{1, \ldots, N_l\}$. Here $Q = Q^T \succeq 0$, $R \succ 0$, $R_\Delta \succ 0$ are weighting matrices and N_p

denotes the length of the prediction horizon.

The solution of the optimization problem (5.22) needs to satisfy the dynamical constraints, expressed by the equality constraints in (5.23). In the same time, other security or performance specifications can be added to the system trajectory. These physical limitations (velocity and bank control inputs) are stated in terms of hard constraints on the internal state variables and input control action as detailed by the set constraints in (5.23). Practically, the sets $\mathcal{X}$, $\mathcal{U}$ denote in a compact formulation the magnitude constraints on states and inputs, respectively. Set $\mathcal{U}_\Delta$ describes the constraints on the variations of the input control signals. In the following, all these sets are supposed to be polytopic (and by consequence bounded) and to contain the reference value. This means that $\xi^{ref}(k) \in \mathcal{X}$, $u^{ref}(k) \in \mathcal{U}$ and $u^{ref}(k) - u^{ref}(k-1) \in \mathcal{U}_\Delta$.

Note that the cost function is designed to minimize the difference between the nominal and the ideal trajectory, whereas the constraints are imposed on the real trajectory. Additionally, at each step of prediction, the current values have to be superposed over the Voronoi decomposition and the best linearization has to be selected.

The trajectory obtained by applying the optimal control u^* computed in (5.22)–(5.23) is "nominal", in the sense that it does not consider either exogenous noises (i.e., the wind) or the state-dependent linearization error (i.e., the term $\beta_j(\xi(k), u(k))$ from (5.14)).

The "real" trajectory is simply the one where we take into account all perturbations:

$$\xi^\circ(k+1) = A_j\xi^\circ(k) + B_j u^\circ(k) + r_j + \beta_j(\xi^\circ(k), u^\circ(k)) + w(k), \tag{5.24}$$

where the bounded perturbation $w(k)$ denotes the wind.

Subsequently, subtracting (5.17) from (5.24) we obtain the tracking error $z(k) = \xi^\circ(k) - \xi(k)$ measuring the difference between real $\xi^\circ(k)$ and nominal $\xi(k)$ trajectories:

$$z(k+1) = A_j z(k) + B_j u(k)^\delta + \beta_j(\xi^\circ(k), u^\circ(k)) + w(k),\ (\xi^\circ(k), u^\circ(k)) \in \mathcal{V}_j, \tag{5.25}$$

where $u^\delta(k) = u^\circ(k) - u(k)$ denotes the difference between "real" and "nominal" control actions.

Considering that the perturbations are bounded,[6] as long as a stabilizable control action

$$u^\delta(k) = \mathcal{K}(\xi(k), \xi^\circ(k)), \tag{5.26}$$

exists, we can guarantee that the real trajectory (5.24) remains in a bounded neighborhood of the nominal one (5.17). Actually, there will exist a sequence of bounded sets which describe the tube:

$$z(k) \in S_k \leftrightarrow \xi^\circ(k) \in \{\xi(k)\} \oplus S_k,\ \forall k \geq 0. \tag{5.27}$$

[6] If the cell $\mathcal{V}_j$ is bounded, then the nonlinear residue $\beta_j(\xi(k), u(k))$ is also bounded.

For LTI dynamics, the choice of (5.26) is simply a gain matrix which makes the closed-loop dynamics stable. Here, due to the switched nature of (5.17) we need also to switch between the gain matrices:

$$\mathcal{K}(\xi(k), \xi^{\circ}(k)) = K_j(\xi^{\circ}(k) - \xi(k)), \ j = 0, \ldots, N_l, \tag{5.28}$$

with each gain K_j stabilizing the pair (A_j, B_j) in (5.17). Then, we obtain the switched system:

$$z(k+1) = (A_j + B_j K_j)z(k) + \beta_j(\xi^{\circ}(k), u^{\circ}(k)) + w(k), \ (\xi^{\circ}(k), u^{\circ}(k)) \in \mathcal{V}_j, \tag{5.29}$$

which, under the assumption of existence of a common Lyapunov function (via a piecewise Lyapunov function [Hovd and Olaru, 2010] or alternatively by imposing constraints on the dwell time between switches [Colaneri, 2009]) is stable.

Of theoretical interest is the computation of the "tube" defined by the sets $S_{j(k)}$. For LTI dynamics, the set $S_{j(k)}$ is constructed to be robust positively invariant in order to minimize the on-line computations. Here the dynamics of the vehicle change whenever the linearization point changes and it may not be possible to find a common robust positively invariant set. In this case, a hybrid structure can be proposed. That is, we compute robust invariant sets for each of the linearized dynamics and change between them (or scaled versions of them, i.e., $\tilde{S}_{j(k)} \ \lambda(k) \ S_{j(k)}$) whenever the linearization point impose it. To summarize, the practical procedure is the following:

- as long as the system use the linearization point $(\xi^{ref}(k), u^{ref}(k))$, the set $S_{j(k)}$ is defined by the same invariant shape and the scalar $\lambda(k) = 1$,
- if the index at step $k-1$ is $j(k-1)$ and $j(k-1) \neq j(k)$, then the shape of $S_{j(k)}$ changes as a consequence of the fact that the linearization index changes.

The scalar $\lambda(k)$ is computed via the LP problem:

$$\begin{aligned} &\lambda(k) = \min \lambda \\ &\text{s.t.} \quad \lambda S_{j(k)} \supset S_{j(k-1)}. \end{aligned}$$

The danger, from the stability point of view, is to have a monotonic increase of the coefficients λ along the switches. This remark provides a simple and efficient criterion for detecting the misfunctioning of the predictive feedback loop: the violation of a pre-imposed bound on the scaling factor. Practically, as long as the change between different dynamics is slow enough, the overall stability of the tracking error, and thus of the boundedness of the tube are preserved due to local contractive properties of each LTI mode (5.29).

Although, the generic robust predictive control strategy would be feasible for the UAV application, we do not pursue here the invariant set manipulation in the implemented

control scheme. We will test the nominal predictive control scheme and show that the inherent robustness of this control law covers in a satisfactory manner the tested maneuvers. We mention however, that whenever the control law needs to pass by certification procedures for covering important wind variations, the robust version of the design needs to be adopted. Moreover, it needs to include the tube description together with a watchdog mechanism for the invariant set scaling factor, which can commute the UAV functioning towards a "safe mode".

Finally, let us recapitulate in Algorithm 5.1 the mechanism implemented based on the theoretical elements presented previously.

Algorithm 5.1: Trajectory tracking optimization-based control problem

Input: Give the collection of way-points $\mathbb{P}$ as in (5.10)

-construct the flat trajectory as in (5.8)–(5.9), passing through the way-points $p^i \in \mathbb{P}$;
-choose a collection of linearization points $\mathbb{L}$ as in (5.13);
-construct the PWA function as in (5.17) with (A_j, B_j, r_j) defined as in (5.15);
-partition the state-space into Voronoi cells as in (5.19);
for $s = 1 : s_{max}$ **do**
    -select the linearization point $l^j \in \mathbb{L}$ by testing (5.20);
    -select the pair (A_j, B_j, r_j) by testing (5.21);
    -find the optimal control action u^* by solving (5.22);
    -compute the next value of the state

$$\xi(k+s+1) = A_j \xi(k+s) + B_j u(k+s) + r_j;$$

end

The forthcoming section provides software-in-the-loop simulations in comparison with actual experimental results on real UAVs, which validate our proposed approach.

5.1.6 Simulation and experimental flight tests results

The flight experiments took place at the Portuguese Air Force Base OTA in May 2012 on the platforms illustrated in Figure 5.1, Figure 5.2, Figure 5.3 owned by the Air Force Academy and the LSTS laboratory of the University of Porto. The results were obtained during a week long stay at the Portuguese Air Force Base OTA linked with the LSTS lab. During this stay, we had more than ten flight tests with different scenarios that evaluated our predictive control algorithm. From these experimentations we have drawn several conclusions. Needless to say, we have encountered difficulties, both numerical and theoretical, but in the end, we consider the results to be of very high performance levels. Moreover, they indicate that the effort of developing such algorithms and investigating formal proofs of convergence and stability is worthwhile.

We show in the rest of the chapter illustrative simulations and ground test results. The control algorithm is running in MATLAB which is interfaced with Piccolo software using DUNE and IMC messaging system described in Section 5.1.1.2.

We proceed to test several scenarios, both with different way-point lists described as in (5.10) and control strategies. The control objective is to force the UAV to track the way-points denoted as red dots in the forthcoming illustrative figures. The control inputs are the velocity and the bank angle of the UAV. One real-world situation that matches this scenario is that of an autonomous aircraft equipped with GPS, radio communications and a camera. The aircraft needs, for example, to take some snapshots at a certain time of a certain area and then, to transmit the information.

Furthermore, to test the proposed trajectory tracking method we use an extended aircraft model of (5.1) (for low-level control) with 12 states, in a 3-DOF simulation. For the real tests we kept the same model for the different UAV platforms that we had at our disposal. The UAVs were restricted to a range of speed between 18 and 25 meters per second and to a maximum bank angle of 0.43 radians. In addition, the simulations and the flight tests took place under various wind conditions. It is assume that the intensity of the wind is bounded for some reasonable values, e.g., a maximum speed of 10m/s (for safety reasons the Portuguese Air Force Academy do not test the UAVs if the wind exceeds $11 \sim 12$ m/s).

A. Simulation results

The numerical data used for the simulated vehicle trajectory tracking are showed in the following table:

In a first stage, using the results in Section 5.1.3 we generated a flat trajectory (depicted in blue in Figure 5.11) starting from the current position of the vehicle and passing through the given way-points (depicted in red dots in Figure 5.11). In a second stage, we used the linearized model (see the linearization procedure in Section 5.1.4) for the control part of the trajectory tracking problem.

Figure 5.11 shows simulated tracking performance in the x-y coordinate frame (i.e., north-east coordinate frame). In this case, the simulations were performed in MATLAB. The vehicle tracking performances for the given reference trajectory (depicted in blue in Figure 5.11) are depicted in green in Figure 5.11. As illustrated in green in the same figure, the constraints on the velocity and bank commands are satisfied. It is well known that an increase in the prediction horizon generally leads to better tracking performances thus, imposing a trade-off between complexity and precision.

Figure 5.12 depicts a 3D simulation of the UAV evolution which tracks the given way-points. In the same figure we represented the projection of the trajectory on the ground. Furthermore, Figure 5.12 shows the flight experimental results for the same scenario.

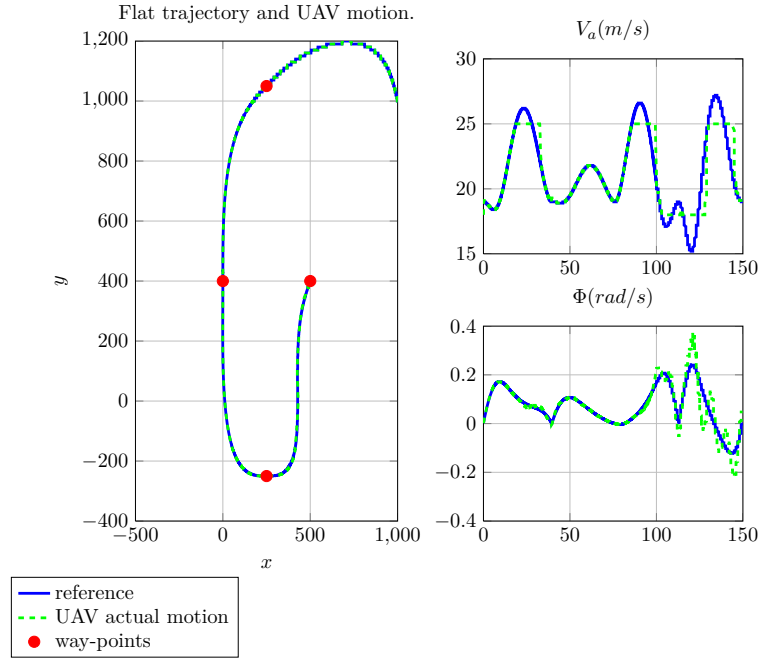


FIGURE 5.11: Reference trajectory and actual UAV motion (simulation).

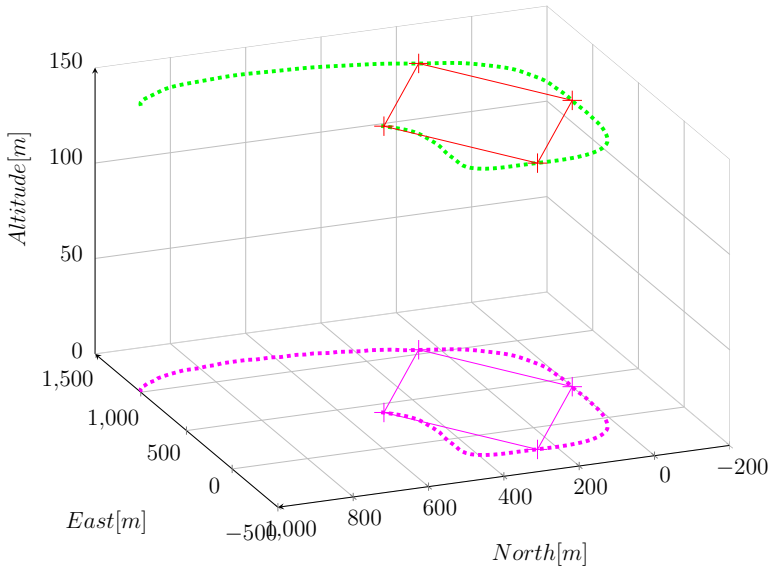


FIGURE 5.12: Actual UAV motion and its projection on the x-y space (simulation).

UAV platform	simulated
Velocity control input	$V_a \in [18,\ 25]$ m/s
Bank angle control input	$\phi \in [-0.43,\ 0.43]$ rad
Variation of the control input signals	$\begin{cases} \text{The variation of } V_a \text{ is limited to } 0.1 \sim 0.2 \text{ m/s}^2 \\ \text{The variation of } \phi \text{ is limited to } 0.5 \sim 1.1 \text{ rad/s} \end{cases}$
Altitude	150 m
Way-points list	$\mathbb{P} = \{(1100, 250, 150), (400, 0, 150), (-250, 250, 150), (400, 500, 150)\}$
Sampling time	100 ms
Tuning parameters	$\begin{cases} Q = [10e1\ 0\ 0; 0\ 10e1\ 0; 0\ 0\ 0.1]; \\ P = [10e2\ 0\ 0; 0\ 10e2\ 0; 0\ 0\ 0.1]; \\ R = 10e4 \cdot [10\ 0; 0\ 1]; \\ R_\Delta = 10e4 \cdot [10\ 0; 0\ 1]; \\ N_p = 7; \end{cases}$
Wind	5 m/s

TABLE 5.1: Simulation results: numerical data specifications.

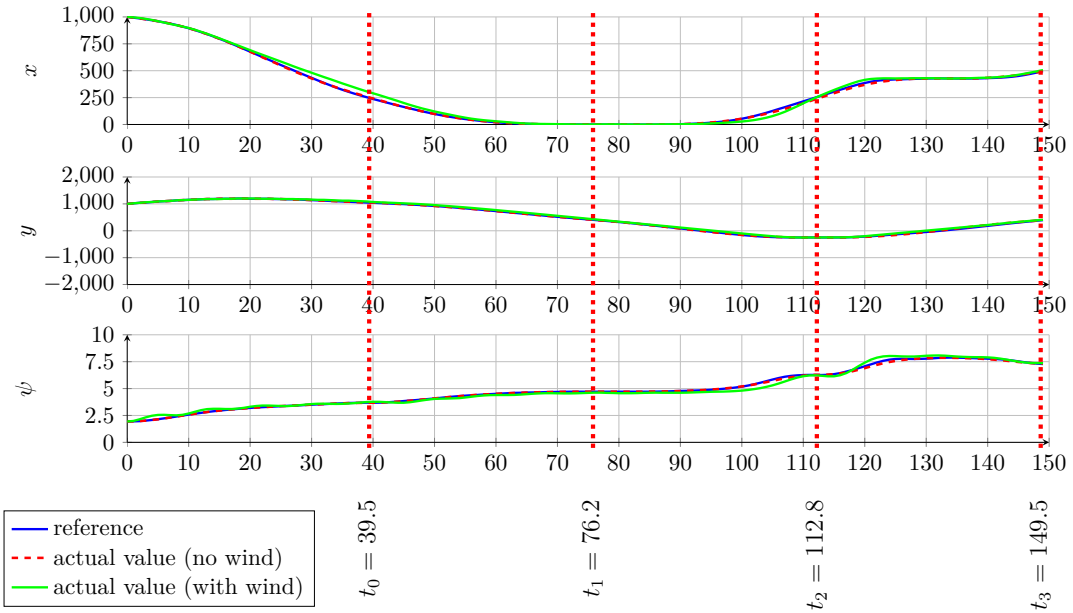


FIGURE 5.13: Comparison between actual and reference UAV motion (simulation).

The predictive tracking controller was also tested in simulations with different wind conditions with a maximum speed of 8 m/s and showed good robustness. Figure 5.13 presents for exemplification the simulated reference trajectory in solid blue and the tracking performance of the UAV with and withought wind conditions. In addition, the figure illustrates the time when the UAV passes through the way-points. The main limitations against superior performance are the wind magnitude and the numerical issues (density of linearization points, prediction horizon length, etc.).

B. Real tests results

We present next the tests results obtained on May 15, 2012, during our stay at the Portuguese Air Force Academy OTA. The numerical data used for the real-time vehicle trajectory tracking are showed in Table 5.2.

UAV platform	"Alfa 06", combustion motor, 2.8 m span, Piccolo II, PC104, 2h30 endurance
Velocity control input	$V_a \in [18\ 25]$ m/s
Bank control input	$\phi \in [-0.43\ 0.43]$ rad
Variation of the control input signals	$\begin{cases} \text{The variation of } V_a \text{ is limited to } 0.1 \sim 0.2 \text{ m/s}^2 \\ \text{The variation of } \phi \text{ is limited to } 0.5 \sim 1.1 \text{ rad/s} \end{cases}$
Altitude	150 m
Way-point list	$\mathbb{P} = \{(0, 600, 150), (300, 0, 150), (0, -600, 150), (-300, -600, 150), (-600, -300, 150), (-300, 0, 150)\}$
Sampling time	100 ms
Tuning parameters	$\begin{cases} Q = [10e1\ 0\ 0; 0\ 10e1\ 0; 0\ 0\ 0.1]; \\ P = [10e2\ 0\ 0; 0\ 10e2\ 0; 0\ 0\ 0.1]; \\ R = 10e4 \cdot [10\ 0; 0\ 1]; \\ R_\Delta = 10e4 \cdot [10\ 0; 0\ 1]; \\ N_p = 7; \end{cases}$
Wind	$5 \sim 7$ m/s to 180 degrees (from South no wind, from West 5.83 m/s), the airplane was flying with the wind.

TABLE 5.2: "Alfa 06" platform: experimental results numerical data specifications.

The flat trajectory generated by the resolution of a system of equation for the reference trajectory at the way-points (see Section 5.1.3) does not take into consideration constraints *in between* the way-points. That is, even if we impose admissible values in the way-points, we can not guarantee what happens in the rest of the trajectory. This was a problem for the real-time experiments, especially for the velocity component of the control input. We had to take into account that the rate of change of the velocity is limited to the maximum acceleration the aircraft can produce, i.e., $0.1 \sim 0.2$ m/s^2.

Strictly speaking, the MPC will need to enforce the constraints satisfaction and thus, we could use the curve provided by the reference trajectory generation mechanism (see the illustrative simulation results in Figure 5.11). It is clear that the existence of such bounds will represent a challenge for the MPC controller, if the generated reference trajectory had some aggressive values on the velocity and the bank. Moreover, the system being nonlinear and the tracking errors too large, the closed loop dynamics may become unstable.

This highlighted the importance of bringing the trajectory into acceptable limits. Hence, we have generated the flat trajectory for a simplified model, by considering $V_a(t)$ in (5.4) to be constant and let the bank angle $\phi(t)$ as the single controllable input. Therefore, the equations (5.8)–(5.8) for the corresponding reference state and inputs are the following:

$$\xi^{ref}(t) = \begin{bmatrix} z_1(t) & z_2(t) & \arctan\left(\frac{\dot{z}_2(t)}{\dot{z}_1(t)}\right) \end{bmatrix}^T, \tag{5.30}$$

$$u^{ref}(t) = \begin{bmatrix} V_a & \arctan\left(\frac{V_a}{g}\frac{\ddot{z}_2(t)\dot{z}_1(t)-\dot{z}_2(t)\ddot{z}_1(t)}{\dot{z}_1^2(t)+\dot{z}_2^2(t)}\right) \end{bmatrix}^T, \tag{5.31}$$

where $t \in [t_0, t_f]$ and V_a denotes the constant velocity. In the same time, we made sure to generate the trajectory by leaving some freedom of movement to the variables $V_a(t)$ and $\phi(t)$ from the constraints point of view. As a consequence, they can be used in real time by the predictive controller to cancel the tracking errors.

Figure 5.14 depicts in blue the reference trajectory the initial position $(-202, -7, 2.86)$ to the final position $(221, 269, 1.57)$, the way-points (the red dots) and the control input signals. Observe that the velocity control input of the reference trajectory is constant and the bank control input varies between acceptable limits. Additionally, the figure shows the derivatives of the control input components of the reference trajectory which vary between acceptable limits.

For the control action we have implemented the MPC mechanism using the model described in (5.17), the one where both, the bank angle and the velocity are varying. As we already mentioned, we have used this construction in DUNE where, the control action applies to a more realistic 12-states model.

Figure 5.15 illustrates the actual UAV motion (depicted in dashed green) in North-East coordinate frame of the real flight test. Acceptable tracking performances for the given reference trajectory (depicted in blue in Figure 5.15) are obtained for the UAV. A 3D illustration of the test scenario together with the actual motion of the UAV is depicted in Figure 5.16, while the tracking error is depicted in Figure 5.17.

Remark 5.4. Note that we have a hierarchy of models for the UAV dynamics. Firstly, we use model (5.1), with V_a a constant, in order to generate the flat trajectory as in (5.30)–(5.31). Secondly, the MPC uses the complete model (5.1) to derive a control action, which is finally applied to a 12-states model of the UAV (or to the Piccolo control UAV

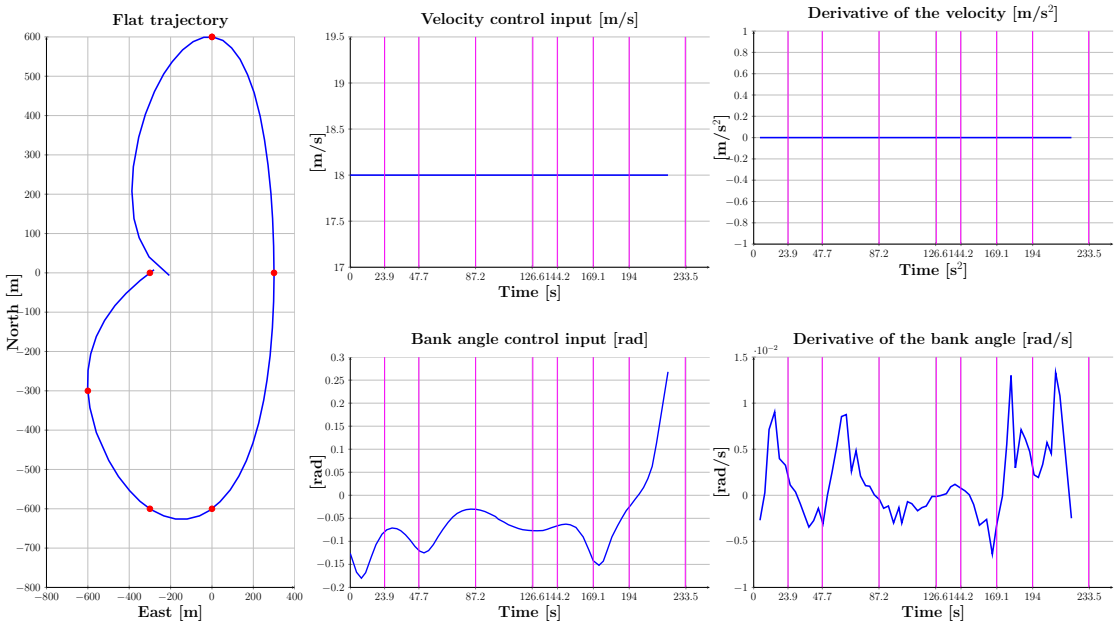


FIGURE 5.14: Reference trajectory, control input signals and their derivatives (flight experiments with Alfa06 UAV).

dynamics during the experiments). Nonetheless, these model mismatches may represent one of the reasons for the tracking error present in recorded test values. However, the predictive controller proves to be robust, in the sense that the error does not increase over time and the real trajectory remains close to the reference. □

In experimental settings we observe that there are losses of communication and the discretization step itself is not constant (see Figure 5.18 where the nominal and actual UAV control motion are illustrated). This rises difficulties in the tracking of the reference trajectory. Figure 5.19 shows the length of the discretization step, depicted in blue. Also, in the same figure, the red line denotes the average length of the discretization step, thus showing that there are values significantly above this average (i.e., the are significant losses of communication during the UAV testing). Observe that the values can get to $10 - 20$ steps more then the normal size (i.e., 0.1 s $-$ 1 s). Some of the negative effects can be mitigated. For example, the generation of the flat trajectory can be considered on-line such that it will always synchronize with the current time step. More precisely, we divided the reference generation into two distinct operation. Firstly, we constructed the continuous representation. At run-time we took the current time and generated the sequence of N_p discretized reference state and input signals starting from the current time. Thus we avoided to generate the entire trajectory beforehand and then, to access indices which would not correspond to the current time. This approach alleviates the

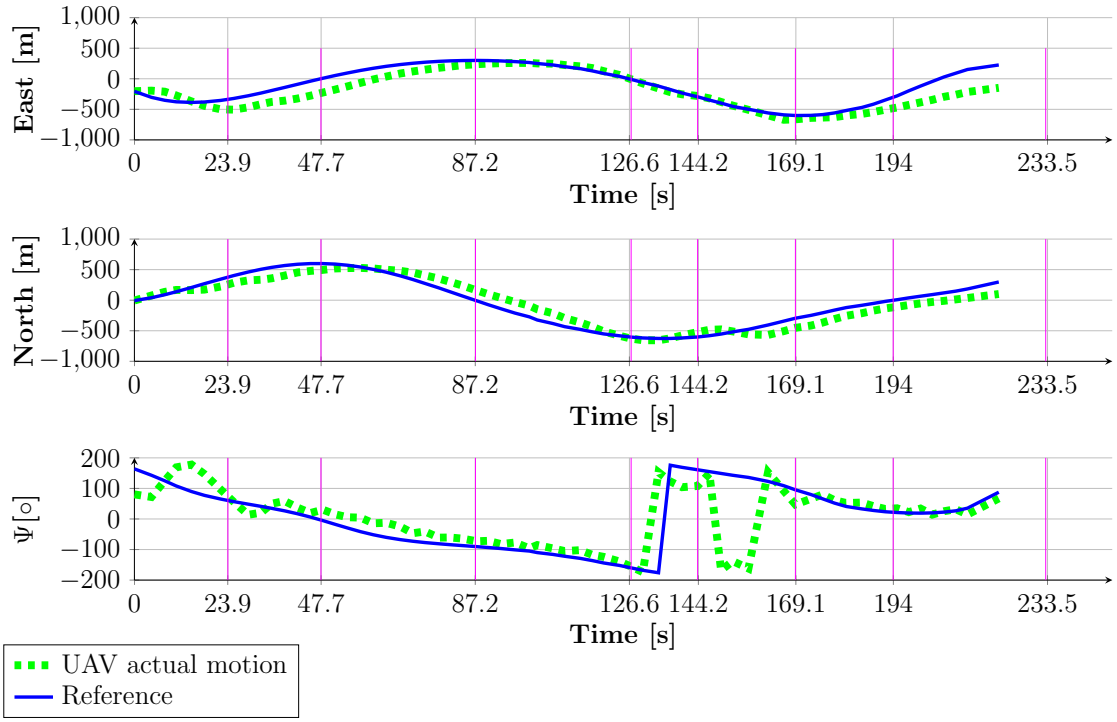


FIGURE 5.15: Reference trajectory and actual UAV motion (flight experiments with Alfa06 UAV).

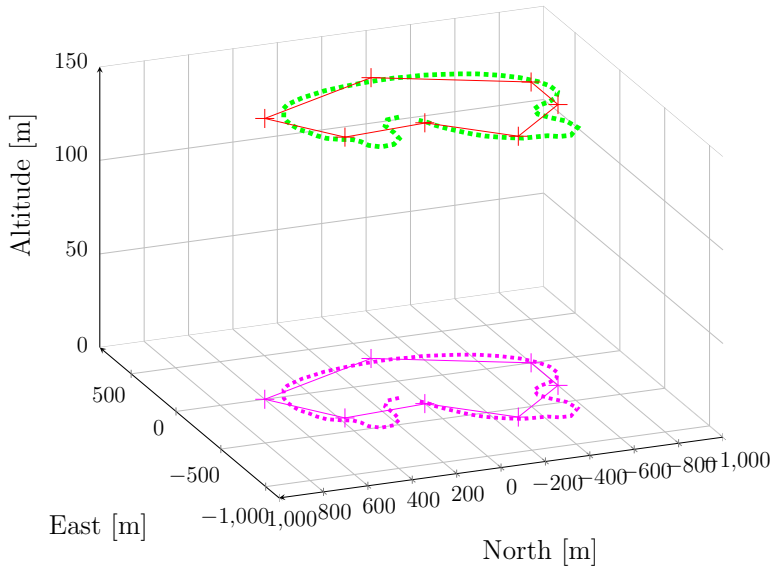


FIGURE 5.16: Actual UAV motion and its projection on the x-y space (flight experiments with Alfa06).

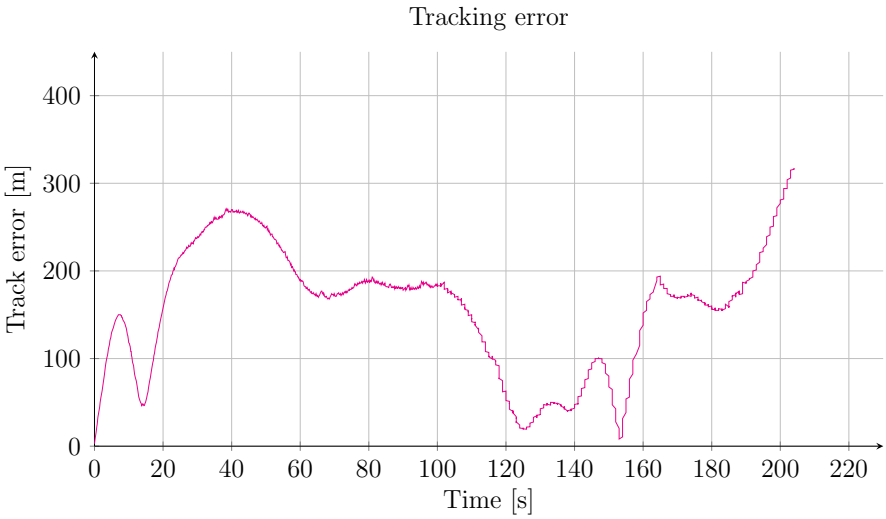


FIGURE 5.17: Tracking error (flight experiments with Alfa06).

possible delays in communication[7]. Also, it may prove helpful for memory usage reasons, instead of having potentially tens of thousands of values, we need only to compute the ones which are necessary at the current step.

Remark 5.5. Lastly, note that we did not considered the effect of measurement errors in the construction of the optimization-based controller. We made use of the MPC implicit robustness for dealing with their effect. □

The wind perturbations were varying between $5 \sim 7$ m/s to 180 degrees (from South to West). In practice, to model the influence of the wind, we considered the returned velocities and compared them with the nominal ones in order to extract the wind value. Consequently we use this value in the cost function (5.22). We realize that this ad-hoc procedure is just a first step in the right direction and points for the future work to an adequate use of an estimation of the wind based on a Kalman filter approach [Østergaard et al., 2007], [Achour et al., 2010].

Algorithm 5.2 recapitulates the mechanism used during the flight tests.

Without entering into extensive details we present next the tests results obtained on May 17, 2012. The illustrative figures show the improvements of the real-time trajectory tracking, due to the small delay in communications and better choice of the tuning parameters. The numerical data used for the real-time vehicle trajectory tracking are showed in Table 5.3. Figure 5.20 depicts the way-points (red dots) and the reference

[7]Note that the tracking error is mostly due to the complex dynamics involved, with significant delays in communications, and does not occur in simulation, see the illustrative figures from the simulation results.

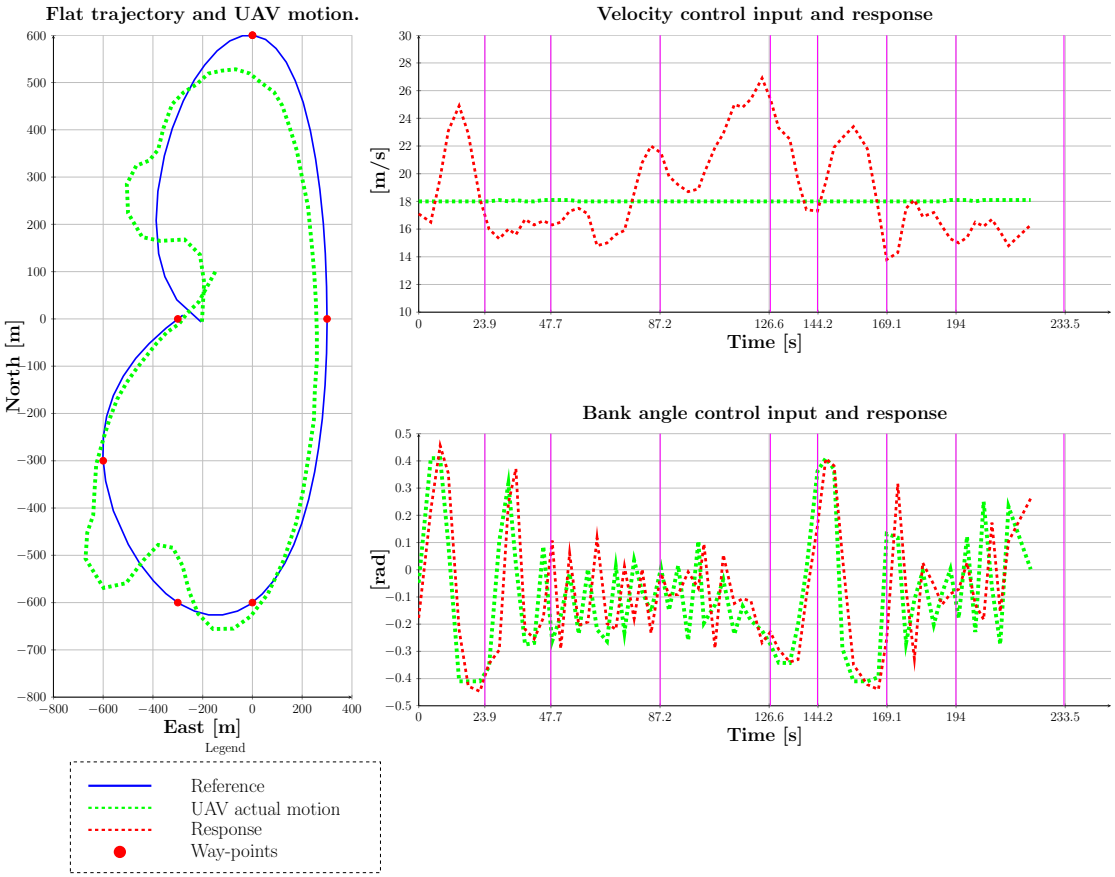


FIGURE 5.18: Nominal and UAV actual motion (flight experiments with Alfa06).

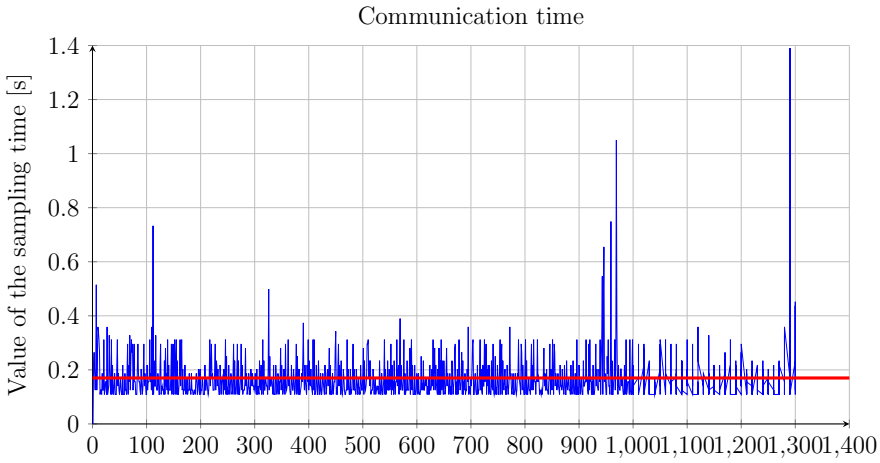


FIGURE 5.19: Value of the sampling time (flight experiments with Alfa06).

Algorithm 5.2: Real-time control of the UAV

Input: Give the collection of way-points $\mathbb{P}$ as in (5.10)

- determine the functions for the reference state and input signals using (5.8)–(5.9) and passing through the way-points $p^i \in \mathbb{P}$;
while *the UAV is flying and is controlled by Piccolo* **do**
  -Piccolo data acquisition;
  if *TRACK==1* **then**
    -take the current time as the starting point for the reference trajectory ($t = 0$);
    -compute references and linearization matrices for the interval $t : t + N_p$;
    -use this information to compute the optimal control action u^* by solving (5.22);
    -compute the next value of the state

$$\xi(k+s+1) = A_j\xi(k+s) + B_j u(k+s) + r_j;$$

  else
    -the UAV is controlled by Piccolo
  end
end

UAV platform	"Pilatos 3", combustion motor, 2.6 m span, Piccolo II, PC104, 1h30 endurance
Velocity control input	$V_a \in [18\ 25]$ m/s
Bank control input	$\phi \in [-0.43\ 0.43]$ rad
Variation of the control input signals	$\begin{cases}\text{The variation of } V_a \text{ is limited to } 0.1 \sim 0.2 \text{ m/s}^2\\ \text{The variation of } \phi \text{ is limited to } 0.5 \sim 1.1 \text{ rad/s}\end{cases}$
Altitude	150 m
Way-points list	$\mathbb{P} = \{(-100, -350, 150), (300, 0, 150), (0, 500, 150), (-300, 500, 150), (-500, 300, 150), (-300, 0, 150)\}$
Sampling time	100 ms
Tuning parameters	$\begin{cases}Q = [10e1\ 0\ 0; 0\ 10e1\ 0; 0\ 0\ 1];\\ P = [10e2\ 0\ 0; 0\ 10e2\ 0; 0\ 0\ 10];\\ R = 10e4 \cdot [10\ 0; 0\ 1];\\ R_\Delta = 10e4 \cdot [10\ 0; 0\ 1];\\ N_p = 7;\end{cases}$
Wind	$4 \sim 6$ m/s to 90 degrees (from West to East), the airplane was flying against the wind.

TABLE 5.3: "Pilatos 3" platform: experimental results specifications.

trajectory (blue line) that Pilatos 3 UAV needs to follow. Furthermore, in Figure 5.21 we illustrate, in the North-East coordinate frame, the actual motion of the UAV (in dashed green). The view of the flight scenario and the UAV motion is represented in a North-East-Altitude coordinate frame in Figure 5.22. Figure 5.24 shows the tracking error which is under 110 m, while Figure 5.18 illustrates the nominal and actual UAV control motion. Observe that the tracking performance is better than in the previous scenario and (as the MPC design framework remains unchanged) is mostly due to the small delayed in communications during the flight test. This can be observed in Figure 5.25 which depicts in blue the length of the discretization step. In the same figure, the average length of the discretization step is represented by the red line, showing acceptable deviations above the average (i.e., the are acceptable losses of communication during the UAV testing).

Remark 5.6. Note that there are small differences between the control algorithm implementation for the practical experiments. In the first we considered for the design of the control action the heading, whereas in the second experiment we used the course. The difference between them is that in the second case we considered a more realistic value of the heading, that is, the course takes implicitly into account the wind. We believe that, besides the small delay in communications, this is another reason for obtaining superior tracking performances. □

5.1.7 Concluding remarks and improvement directions

The present section addresses a Model Predictive Control (MPC) strategy for Unmanned Aerial Vehicles (UAVs). The presented results of software-in-the-loop simulations and real flying tests confirmed the viability of the proposed approach. Moreover, the reference generation proved to be a valuable tool on the predictive control setup, the flat trajectory being an important element towards the constraints verification. Also, the proposed trajectory generation mechanism takes into account way-point conditions and furthermore, allows to obtain linearizations of the nonlinear vehicle model along the flat trajectory.

As mentioned, we have encountered difficulties, both numerical and theoretical. During the presentation we gave solutions to some of them, but of course there is always room for improvements.

Due to the difficulties of the schema (nonlinear dynamics and the associated computational load) we have not implemented the tube MPC approach yet. For the future we propose to construct sets which bound the tracking error and use them in the tube MPC description. Also, investigating formal proofs of convergence and stability is worthwhile.

We also note the need for a better mechanism for wind estimation in order to decrease the uncertainty in the prediction model (e.g., a Kalman filter implementation). Nonetheless, with all these shortcomings and with various (and usually unfavorable) wind conditions

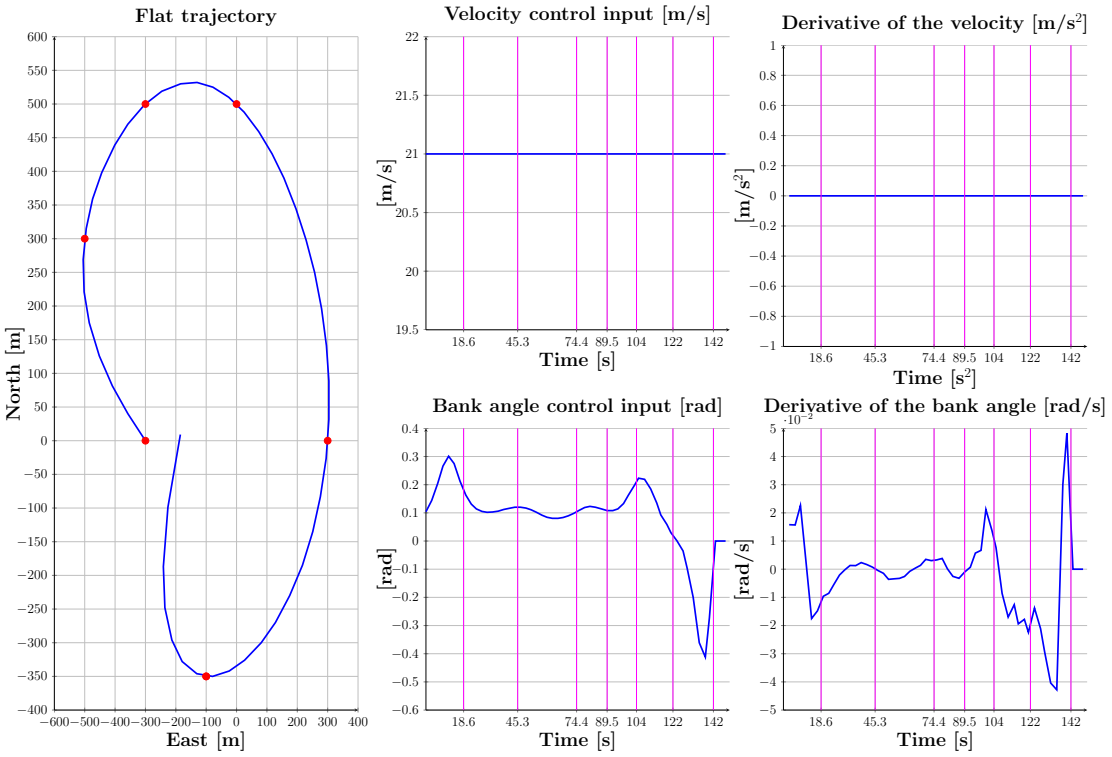


FIGURE 5.20: Reference trajectory, control input signals and their derivatives (flight experiments with Pilatos 3).

the flights tests exhibited an acceptable tracking performance. The implicit robustness of the MPC proved its quality and the results were satisfactorily.

Another sensitive point is the discretization and the inter-sample variations. For now, we assumed a fixed length of the discretization step and computed the input action accordingly. A future improvement of the control mechanism will be the inclusion of these communication delays in the design of the predictive controller.

There are many challenging applications for the multi-agent formations. These challenges often involve the realization of basic behaviors, such as trajectory tracking, formation keeping while avoiding collision, allocating tasks, communication, coordinating actions and team reasoning. In the present section we resumed the real-time flight experiments which did not include multi-agent scenarios for safety considerations. In the next section we will use all the theoretical elements from the previous chapters in order to provide simulation results for trajectory tracking of multi-agent formation. Albeit the agents are described by linear dynamics, we are confident that the results can be extended for more realistic systems as the one used for modeling the UAVs.

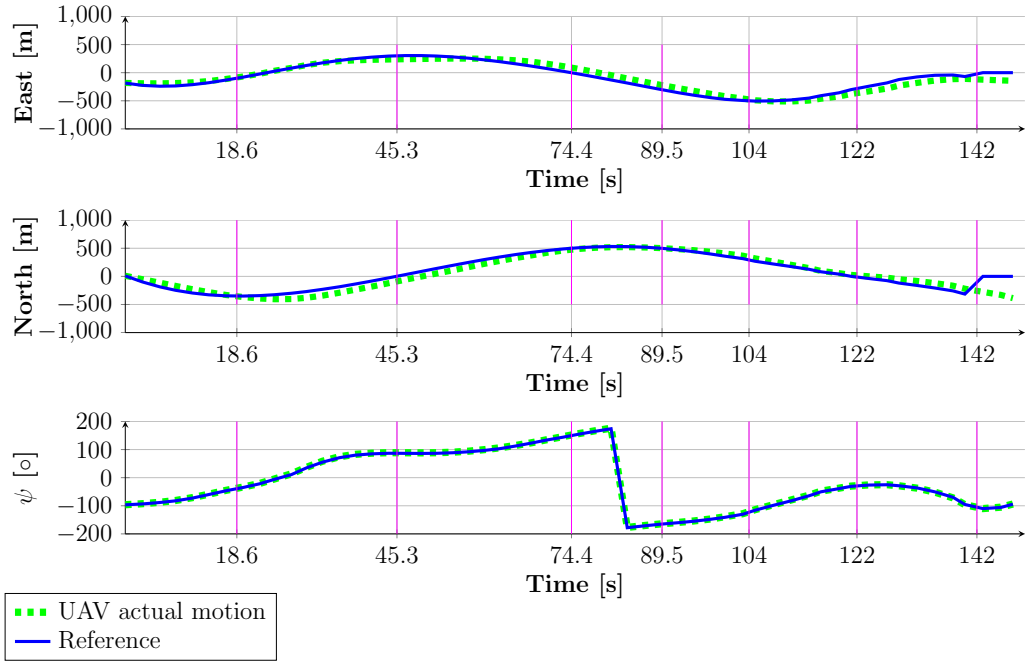


FIGURE 5.21: Reference trajectory and actual UAV motion (flight experiments with Pilatos 3).

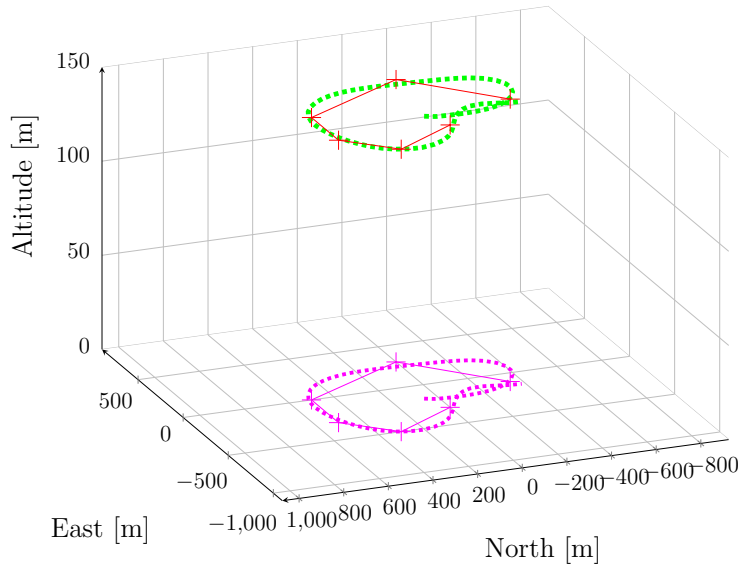


FIGURE 5.22: Actual UAV motion and its projection on the x-y space (flight experiments with Pilatos 3).

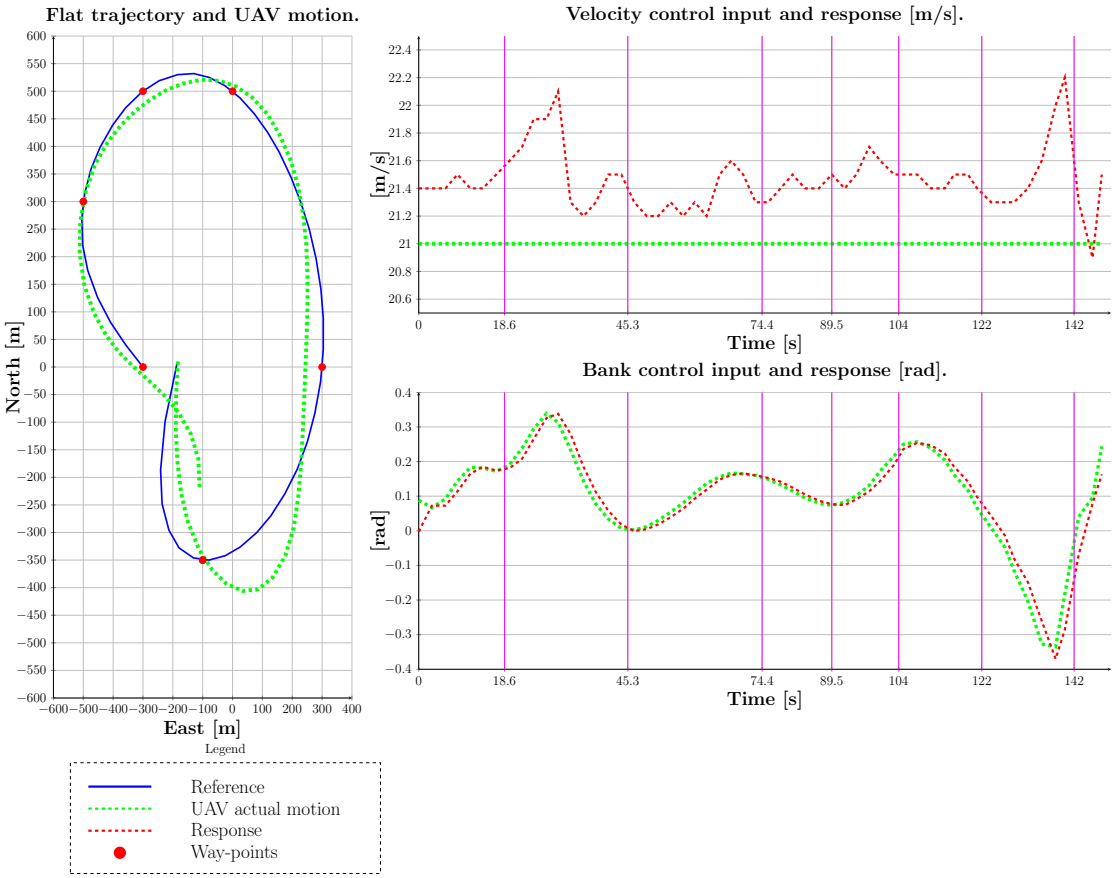


FIGURE 5.23: Nominal and actual UAV motion (flight experiments with Pilatos 3).

5.2 Trajectory tracking for multi-agent formation

In Chapter 3 of the present manuscript we characterized the formation and assured the convergence of multiple agents to a specific configuration. Moreover, we have given a thorough analysis of the control design method which enabled the stabilization of the multi-agent formation. Having all these elements well established, we will concentrate in the present section on the basic task of controlling some agents to follow a predefined trajectory, while maintaing the desired formation pattern. Its outcomes offer a wide range of applications such as security patrols, search and rescue in hazardous environments, area coverage and reconnaissance in military missions. Furthermore, the formation control does not restrict itself only to ground vehicles or robots. It can be applied to aircrafts, especially UAVs, spacecrafts, surface vessels or underwater vehicles. For more examples of applications of multi-agent formation, the reader is referred to the introduction chapter of the present manuscript. In this section we will not discuss the

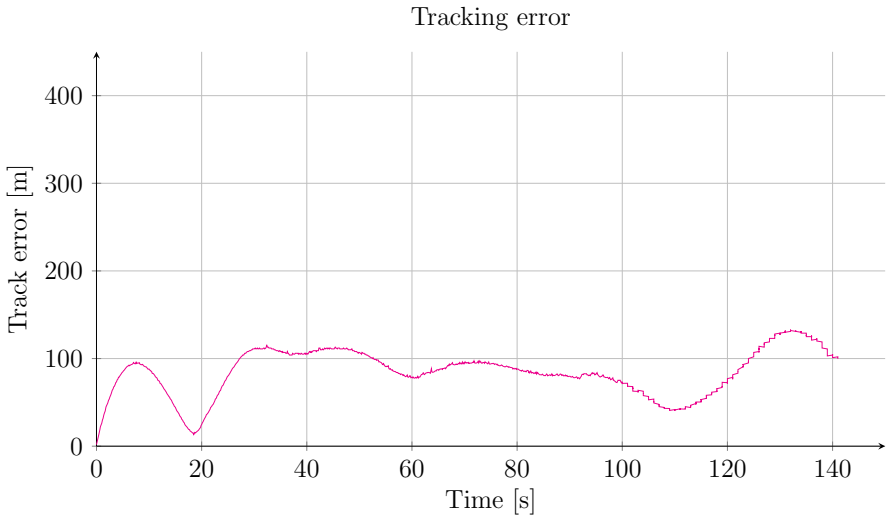


FIGURE 5.24: Tracking error (flight experiments with Pilatos 3).

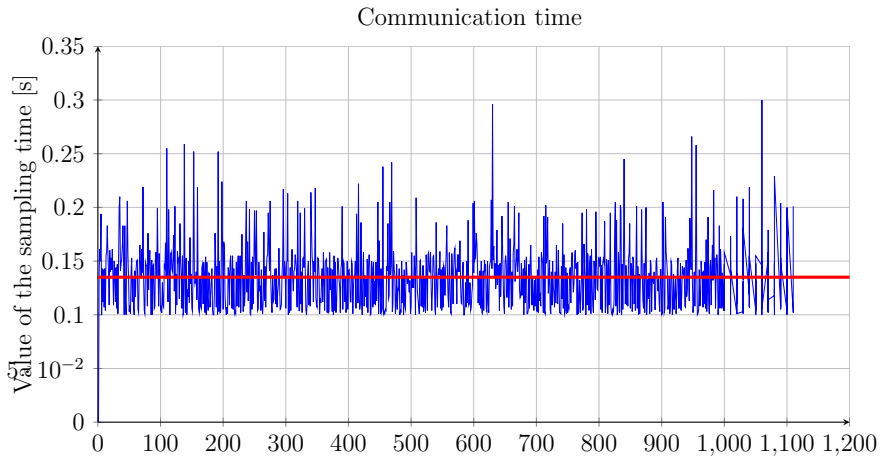


FIGURE 5.25: Value of the sampling time (flight experiments with Pilatos 3).

theoretical elements as they are already established and well detailed in the other chapters. Rather, we will use them to solve the trajectory tracking issue for a formation. We will validate our remarks and observations through simulation and illustrative examples.

We consider only the nominal system as in (3.3) for describing the agents in the tracking problem. However, for each agent we determine a safety region characterized by an invariant set as described in (3.9). The state constraints are represented by the collision avoidance between the agents and/or with obstacles (see Section 3.2, Chapter 3). Besides maintaing a safety distance between the agents we will be also interested in imposing

velocity constraints for an agent. These can represent, for example safety limits, such as a minimum maneuvering velocity near an obstacle or another agent. Another example involves UAVs formation flying, where each agent has to keep its velocity grater than a specified value, even if the formation follows a trajectory with a relative velocity inferior to some pre-imposed bounds for each UAV.

Having these elements, the trajectory tracking problem formulated in a non-convex constrained MPC framework is described from both the centralized and decentralized point of view as a receding horizon mixed-integer optimization problem. Using the predicted control laws, the agents move in the same direction following a specified trajectory.

Based on the information received from the MPC formulation the constraints are taken into account, leading the agents to follow the reference trajectory in a formation which depends on the geometry of the constraints.

The specified trajectory of the group of agents is generated using the differential flatness formalism already described in Chapter 2 and applied for the real-time control of an UAV in the previous section. Finally, we will show that we can achieve a group formation only by imposing constraints on the position and the velocity of the agents, while they follow an a priori given reference.

In order to have a clear image of the elements needed for solving the trajectory tracking problem for multi-agent formation, we recall through a simple illustration, Figure 5.26, the steps taken in the off-line and on-line implementation.

For the optimization problem we recall here the cost function $V_n(\mathbf{x}, \mathbf{u}) : \mathbb{R}^{N_a \cdot n} \times \mathbb{R}^{N_a \cdot m} \rightarrow \mathbb{R}$ which aims at maintaining the formation, following the reference trajectory. The centralized optimization problem under collision avoidance constraints is formulated as[8] in Section 3.4.2, Chapter 3:

$$\mathbf{u}_{\mathcal{I}}^{*} = \arg \min_{\mathbf{u}_{\mathcal{I}}(k|k),\ldots,\mathbf{u}_{\mathcal{I}}(k+N_p-1|k)} V_n^{ref}(\mathbf{x}_{\mathcal{I}}(k|k), \mathbf{u}_{\mathcal{I}}(k|k), \ldots, \mathbf{u}_{\mathcal{I}}(k+N_p-1|k)) \tag{5.32a}$$

$$\text{subject to:} \quad \begin{cases} \mathbf{x}_{\mathcal{I}}(k+s+1|k) = \mathbf{A}_{\mathcal{I}}\mathbf{x}_{\mathcal{I}}(k+s|k) + \mathbf{B}_{\mathcal{I}}\mathbf{u}_{\mathcal{I}}(k+s|k), & s = 0, \ldots, N_p - 1, \\ \mathbf{x}_{\mathcal{I}}(k+s|k) \notin \mathbb{S}\ \forall i, j \in \mathcal{I},\ i \neq j,\ s = 1, \ldots, N_p, \end{cases} \tag{5.32b}$$

with $x_{\mathcal{I}}(k|k)$ and $u_{\mathcal{I}}(k|k)$ the corresponding vectors which collects the states and the inputs of each individual nominal system (3.3) at time k. The cost function, is defined

[8] To emphasize the presence of the reference inputs and states in the formulation of the cost function we have added the ref superscript.

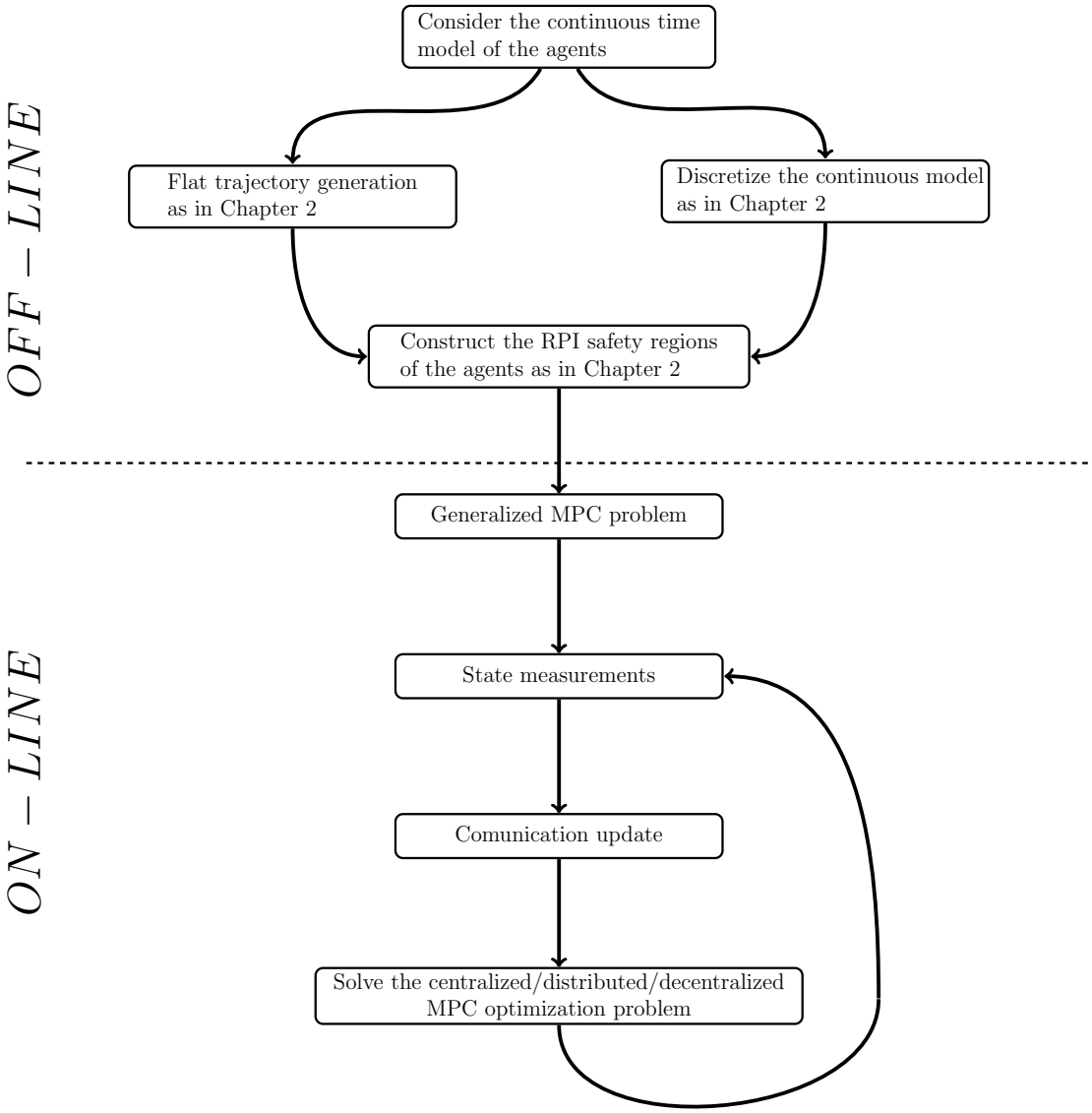


FIGURE 5.26: General MPC approach for trajectory tracking of multi-agent formation.

as in (3.32):

$$V_n^{ref}(\mathbf{x}_{\mathcal{I}}(k|k), \mathbf{u}_{\mathcal{I}}(k|k), \ldots, \mathbf{u}_{\mathcal{I}}(k+N_p-1)|k)) = \|\mathbf{x}_{\mathcal{I}}(k+N_p|k) - \mathbf{x}_{\mathcal{I}}^{ref}(k+N_p|k)\|_{\mathbf{P}} + \tag{5.33}$$

$$+ \sum_{s=1}^{N_p-1} \|\mathbf{x}_{\mathcal{I}}(k+s|k) - \mathbf{x}_{\mathcal{I}}^{ref}(k+s|k)\|_{\mathbf{Q}} + \sum_{s=0}^{N_p-1} \|\mathbf{u}_{\mathcal{I}}(k+s|k)^T - \mathbf{u}_{\mathcal{I}}^{ref}(k+s|k)\|_{\mathbf{R}},$$

where $(\mathbf{x}_{\mathcal{I}}^{ref}(k|k), \mathbf{u}_{\mathcal{I}}^{ref}(k|k))$ represents the reference trajectory to be followed by the agents at current time.

In the following we present simulation examples in order to validate the approaches presented thought the manuscript.

5.2.1 MIP-based solution for trajectory tracking

Consider for the agents with a double integrator (3.10) models, where $[x_i \ y_i \ v_{x,i} \ v_{y,i}]^T$, $[u_{x,i} \ u_{y,i}]^T$ are the state and the input of each system. The components of the state are: the position (x_i, y_i) and the velocity $(v_{x,i}, v_{y,i})$ of the i^{th} agent, $i \in \mathcal{I}$, with $\mathcal{I}$, the collection of all agents indices as in (3.1).

- Trajectory tracking of multi-agent formation with collision avoidance constraints – centralized MPC approach

This example considers three homogeneous agents (depicted as blue, green and red dots, respectively in Figure 5.27) following the same trajectory (depicted in magenta). The collision avoidance constraints are imposed by the safety regions associated to each agents. Note that since the agents are homogeneous, the safety regions are the same and they are represented by the blue, green and red bounded polyhedral sets in Figure 5.27. Each safety region points in the direction of each agent velocity vector. Figure 5.27 illustrates at three different time instants the evolution of the agents. In order to avoid the collision and according to the optimization result, the agents are self-organized (and can be assimilated with a flocking behavior) into the configuration depicted in the same figure.

The tracking and the collision free maneuvering for the given reference trajectory is achieved with a prediction horizon $N_p = 3$. Figure 5.28 illustrates the evolution of the three heterogeneous along the reference trajectory. Since the goal is to steer the agents toward the reference trajectory we choose cost matrices which penalize only the position component of the state (the first 2 states of the vector state) $\mathbf{P} = 500 \cdot \text{blkdiag}(I_2, O_2, I_2, O_2, I_2, O_2)$, $\mathbf{Q} = 10 \cdot \text{blkdiag}(I_2, O_2, I_2, O_2, I_2, O_2)$, $\mathbf{R} = 1$. It can be observed that once the transitory phase ends and the formation is achieved, its center of the mass is tracking the reference as a unitary system. Inside the formation, the relative order remains unchanged (as it can be seen in Figure 5.27), the blue agent remaining at the left extremity of the xy plane irrespective of the heading of the trajectory. This confirms that the formation does not exhibit, a leader-follower behavior, aspect which is coherent with the homogeneity of the weightings in the MPC problem. In the MIP formulations, imposed by the collision avoidance constraints, the computation time is in the worst case scenario exponentially dependent on the number of binary variables used. In this particular example, only two binary variables suffices for describing the feasible

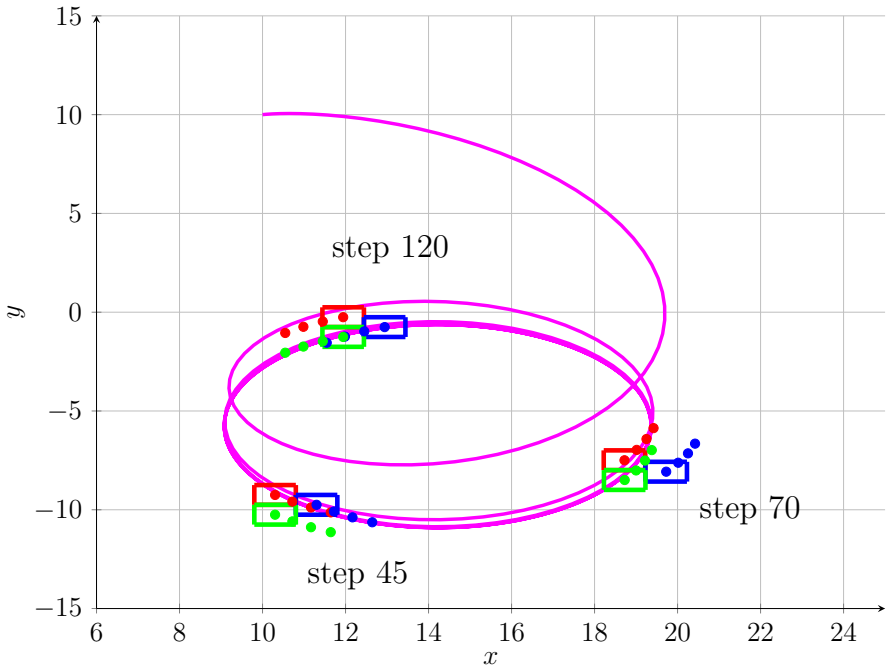


FIGURE 5.27: Actual motion of 3 homogeneous agents in a triangle formation with collision avoidance constraints at different time instances.

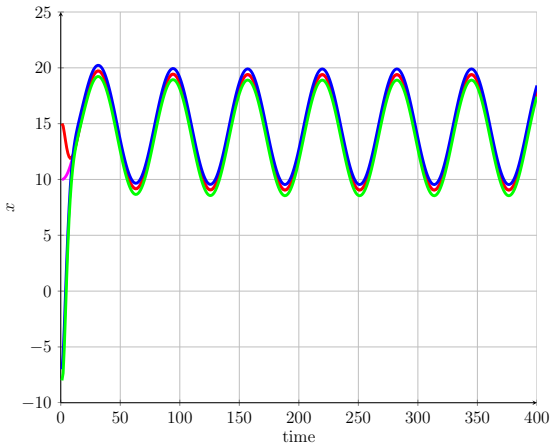
region, thus the problem is tractable even for a prediction horizon grater then the one we used, $N_p = 3$.

- Trajectory tracking of multi-agent formation with collision avoidance and velocity constraints – centralized MPC approach

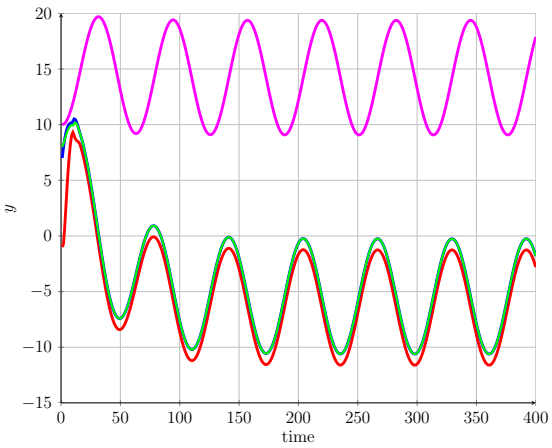
We go further and consider the same 3 homogeneous agents both with collision avoidance and velocity constraints. A simple velocity constraint would be to keep the speed magnitude above some minimal value, v_{min}. This is a natural demand for UAVs for example, where a too small speed will loose the portability and ultimately crash the airplanes. The minimal speed condition can be approximated by a polyhedral norm condition, i.e., $\|v\| \geq v_{min}$, practically expressed by the non-convex constraints (expulsion type):

$$\begin{aligned} v_{x,i} \leq -v_{min}, \quad \text{or} \quad -v_{x,i} \leq -v_{min}, \quad \text{or} \\ v_{y,i} \leq -v_{min}, \quad \text{or} \quad -v_{y,i} \leq -v_{min}, \end{aligned} \tag{5.34}$$

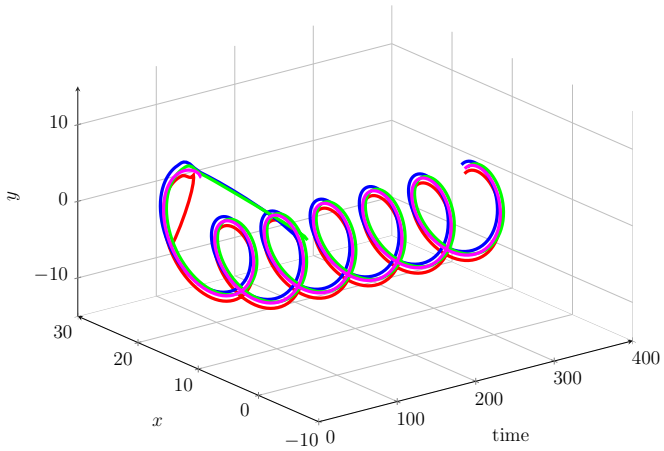
where the constant $v_{min} > 0$. These constraints are rewritten in a linear form using MIP formulation in the case of the collision avoidance constraints. In this particular example we choose $v_{min} = 8$ m/s.



(A) X-axis



(B) Y-axis



(C) X,Y-axis

FIGURE 5.28: Reference trajectory and the time evolution of the 3 homogeneous agents in a triangle formation.

Figure 5.29 shows how the agents are self-organizing in a triangle formation and the center of mass (denoted by the asterix) of the formation follows the reference trajectory (depicted in magenta in the same figure). Figure 5.30 illustrates the good tracking performances of the center of mass (in blue), for a prediction horizon as low as $N_p = 3$.

For different initial conditions and tunning parameters of the optimization problem, the simulations show that the agents have a regular motion (once the formation is achieved, stage which is notified by the activation of the anti-collision constraints) while following the path in a specific formation. The state constraints are always satisfied.

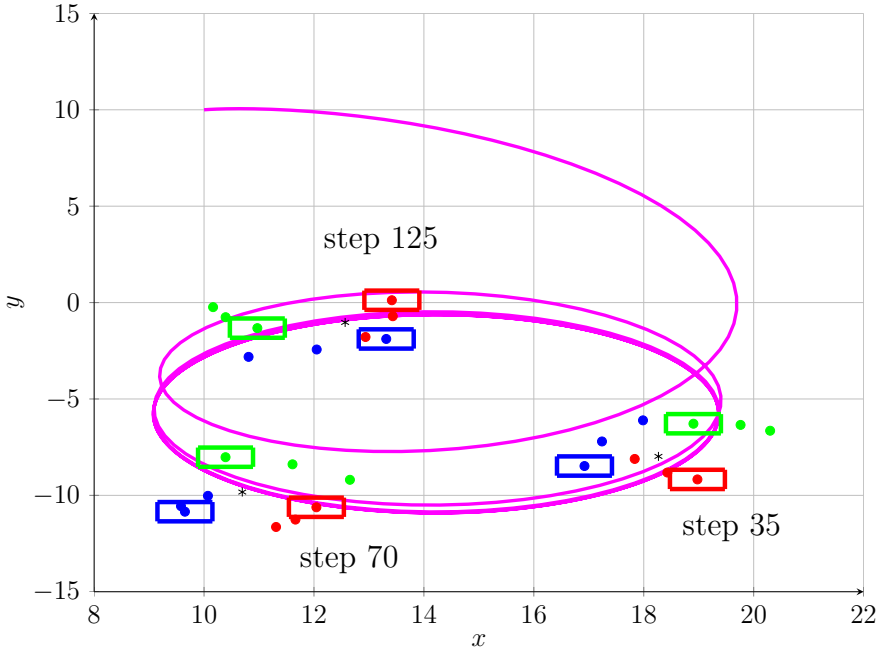
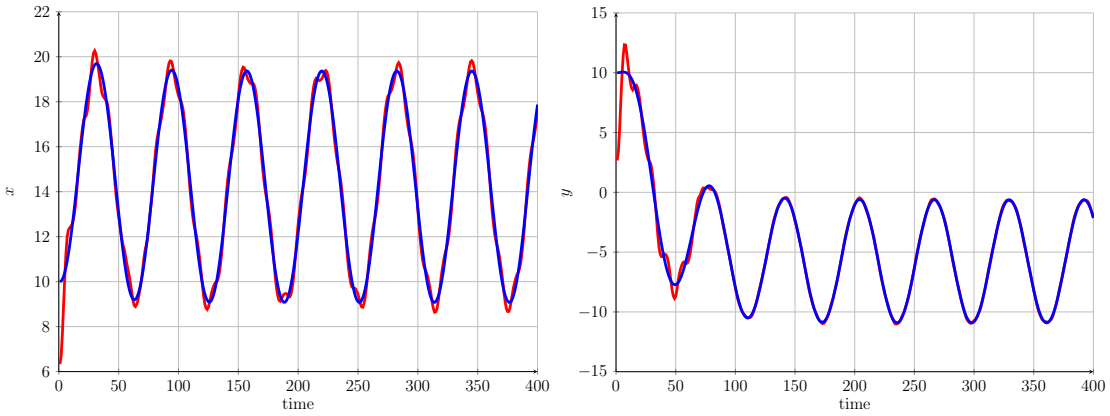


FIGURE 5.29: Actual motion of 3 homogeneous agents in a triangle formation with collision avoidance and velocity constraints at different time instances.

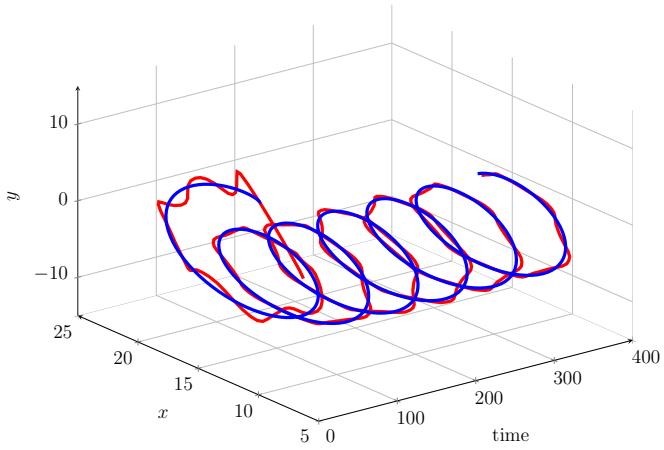
- Trajectory tracking of multi-agent formation with collision avoidance constraints – decentralized MPC approach

Following the decentralized procedure proposed in Section 3.6.1, Chapter 3 in this example we consider the global model of the 3 homogeneous agents from previous examples, decomposed in 3 subsystems for the prediction purposes. The control action for agent one is computed with the information from the pair (agent 1, agent 2), the control action for agent 2 ←(agent 2, agent 3) and the control action for agent 3 ←(agent 3, agent 1). The agents of each subsystem are imposing a relative distance at the local MPC design level. The prediction horizon is $N_p = 2$, with the remark that increasing the prediction horizon leads to a better tracking of the reference trajectory but, obvious increase in the computational time die with the number of constraints to be handled via MIP techniques. The reference tracking is achieved (Figure 5.31) by the 3 homogeneous agents without collision and preserving the safety distance between agents as imposed by the cooperative constraints illustrated in Figure 3.14, Chapter 3.

It is important to point out that, due to the conservative constraints imposed in the decentralized case, the formation structure is different from the centralized one with a degradation of the overall tracking performance. Figure 5.32 illustrates each agent



(A) X-axis (B) Y-axis



(C) X,Y-axis

FIGURE 5.30: Reference trajectory and the time evolution of the center of mass of a triangle formation with 3 agents.

trajectory along the reference. Note that the blue agent acts like a leader and is tracking very well the trajectory.

Up to now we have considered for the optimization problem the combination of MPC with MIP for providing the control action. In the following, we consider the Potential Field-based solution given in Section 3.6.2, Chapter 3.

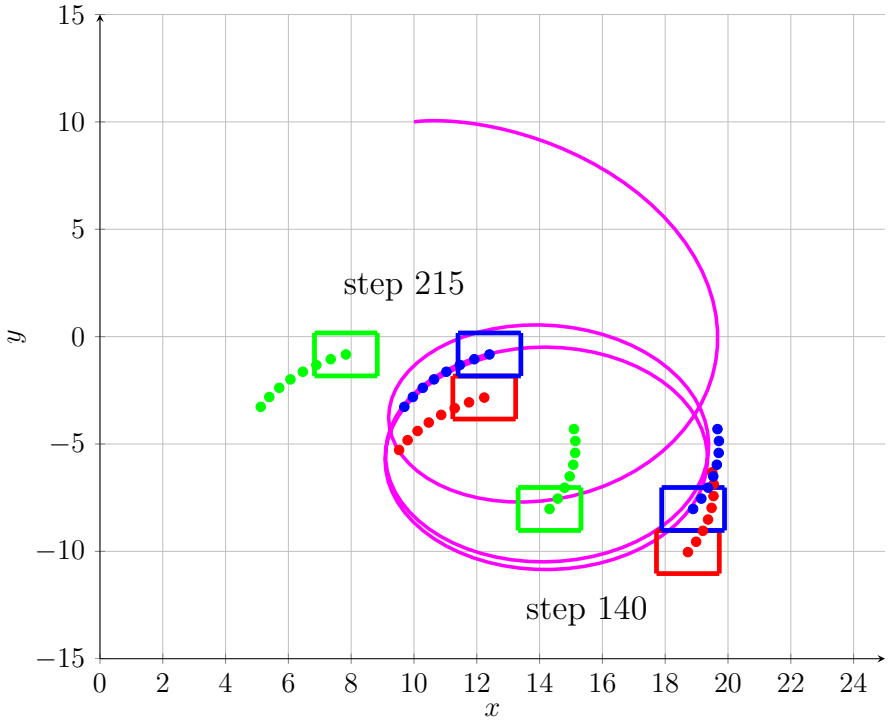


FIGURE 5.31: Actual motion of 3 agents in formation with collision avoidance constraints at a certain time instance (decentralized MPC case)

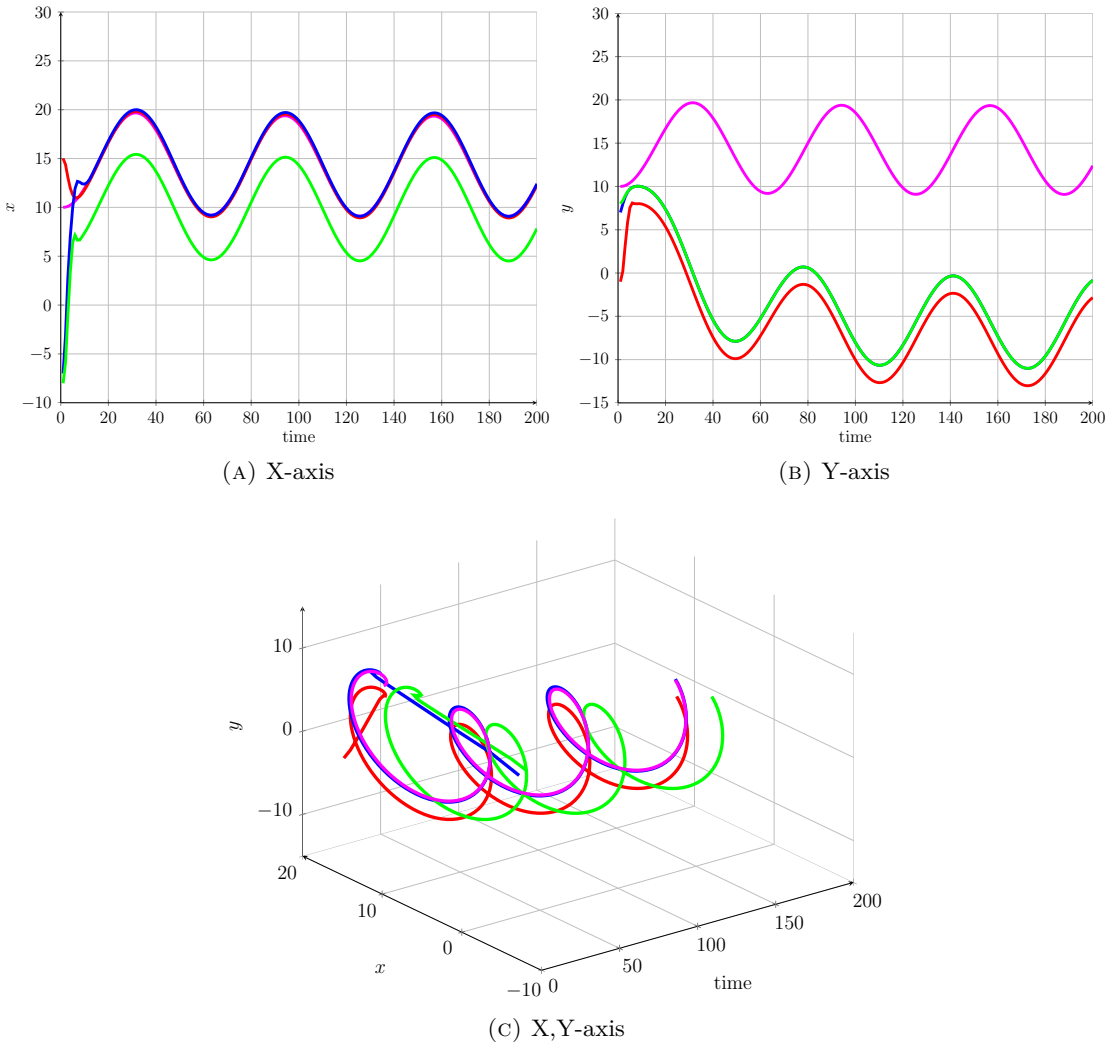
.

5.2.2 Potential Field-based solution for trajectory tracking

In the following simulation examples we keep the same double integrator dynamics for the agents and provide illustrative results for the formation control in the presence of collision avoidance constraints.

- Trajectory tracking of a leader/followers formation with collision avoidance constraints – decentralized MPC approach

Consider five homogeneous agents (i.e. $N_a = 5$ in (3.10)) described by the dynamics (3.10), with $m_1 = 45$kg, $m_2 = 60$kg, $m_3 = 30$kg, $m_4 = 50$kg, $m_5 = 75$kg, $\nu_1 = 15$Ns/m, $\nu_2 = 20$Ns/m, $\nu_3 = 18$Ns/m, $\nu_4 = 35$Ns/m, $\nu_5 = 23$Ns/m. The initial positions and velocities of the agents are chosen randomly. A polyhedral safety region as in (3.9) is associated to each agent. We take arbitrarily $l = 1$ to be the leader which has to be followed by the rest of the agents $i = 2, \ldots, 5$ $(i \neq l)$. For the leader dynamis we generate through flatness methods, state and input references and for both types of agents we use



(A) X-axis

(B) Y-axis

(C) X,Y-axis

FIGURE 5.32: Reference trajectory and the time evolution of the 3 agents in a triangle formation (decentralized case).

MPC in order to construct the control action. A quadratic cost function as defined in (5.32) is used for the leader with a prediction horizon $N_{p,l} = 10$.

For the followers we consider a potential function as the cost function in the optimization problem (3.54), with a prediction horizon $N_{p,f} = 2$. The potential will be constructed such that both the following of the leader and the maintainance of a formation are achieved. The neighborhood radius is set to $r = 8$ m, the weighting coefficients are $\beta_r = 1$, $\beta_a = 10$, $c_3 = 1$, $c_4 = 0.25$, $\beta_v = 15$. The effectiveness of the approach is confirmed by the simulation depicted in Figure 5.33, where the evolution of the homogeneous agents is

represented at three different time instances. The agents successfully reach a formation and follow the leader without trespassing each other safety regions.

Is worth to be mentioned that the prediction horizon needs to be kept as low as possible with a difference in between the leader and the followers (smaller for the followers than the one used for the leader). This is justified by the fact that the trajectory tracking becomes a principal task for the leader. For the followers, additional prediction step for the cost function (which is not quadratic) incurs significant computational complexity.

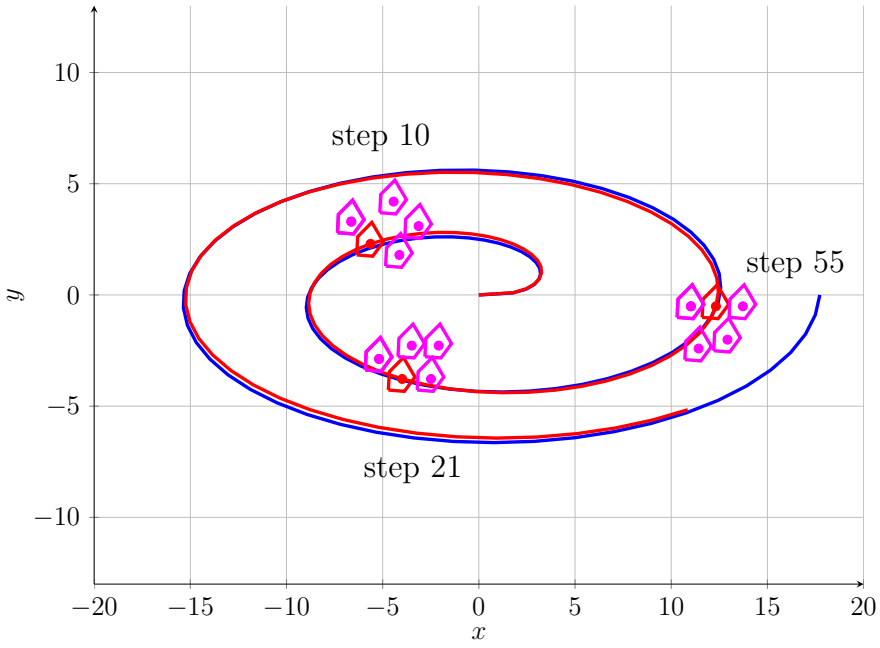


FIGURE 5.33: Trajectory tracking of the leader/followers formation at different time instances with their safety regions (leader in red, followers in magenta).

- Trajectory tracking of a leader/followers formation with collision and obstacle avoidance constraints – decentralized MPC approach

We build upon the previous example and we consider additionally obstacle avoidance. Furthermore, we redesign the reference trajectory such that it avoids stationary and a priori known obstacles. More precisely we add way-points n the flatness design, which steer the reference trajectory from the interdicted region.

In Figure 5.34 the original reference (in blue) illustrates the flat trajectory which does not take into account the obstacle. On the other hand, by adding an additional control point we were able to construct a trajectory which avoids the obstacle (in red). Satisfactory

tracking performances for the given reference trajectory are obtained with a prediction horizon $N_{p,l} = 10$, as well as in the previous example.

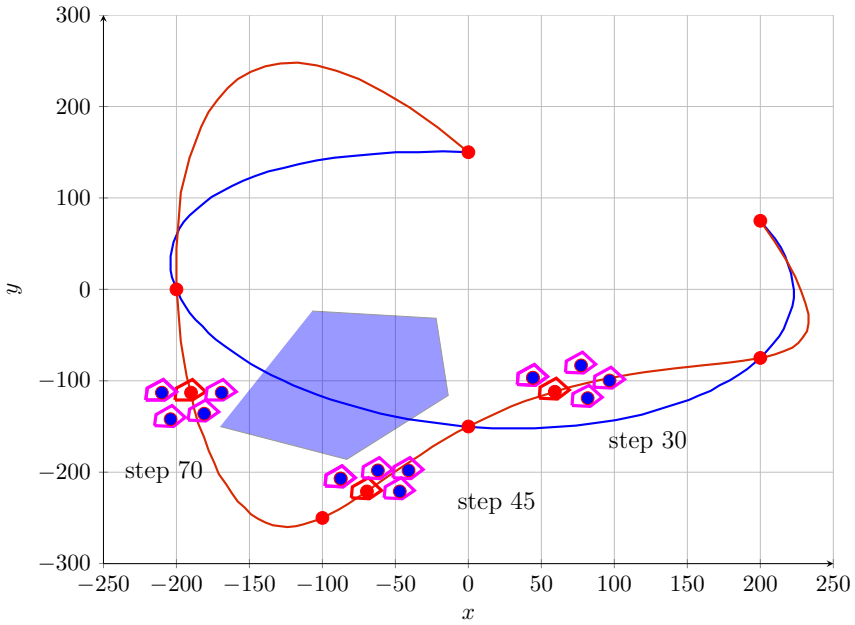


FIGURE 5.34: Obstacle avoidance and trajectory tracking of the leader/followers formation at different time instances with their safety regions (leader in red, followers in magenta).

5.2.3 Concluding remarks

We have shown various simulation results for tracking problems of multiple agents, under different formulation and optimization algorithms spanning from MIP based formulation to Potential field in order to deal with collision avoidance requirements. While the MIP lead to tight formation and additional information of the sensitive maneuvering due to the available status of constraints activation, the Potential field approach has the advantage of a very good scaling of the complexity with the prediction horizon.

Finally, is worth mentioning the degrees of freedom offered by the flatness based trajectory generation, which relieves the feedback control law by providing a less aggressive trajectory with respect to the static obstacles collision avoidance.

Chapter 6

Conclusions and future developments

6.1 Conclusions

The present manuscript had as main objective to develop and to shed light on the optimal control of multi-agent dynamical systems in the presence of constraints. Elements from control theory, optimization and computer science have been merged together and have provided us with useful tools that were further applied to different aspects of the problems involving multi-agent formulations. In particular, we have concentrated on a geometrical interpretation for the control and coordination of multiple agents. To this end, we have made use and built upon a combination of optimization-based control and set-theoretic tools.

We have to mention that at the start of this research topic investigation, the domain of cooperative multi-agent systems was "terra incognita" to us. Moreover, the sheer size of the area makes an exhaustive analysis impossible. Consequently, we have chosen a specific direction of research (in line with our previous background) and tried to pursue it to the best of our ability. Hence, we do not claim that the contributions presented through the manuscript are revolutionary but rather evolutionary. Nonetheless, we hope and advocate that the well established concepts from set-theoretic methods, differential flatness, Model Predictive Control (MPC), Mixed-Integer Programming (MIP) can be adapted and sometimes enhanced for solving some challenging problems appearing in the control of cooperative systems.

With respect to related works, we have seen a great variance in the actual definition of the "agent" notion. Heterogeneous entities in cooperative systems could be as simple as scalar agents without any dynamics or second-order point-mass models or even nonlinear dynamical systems with nonholonomic motion constraints. For our study we

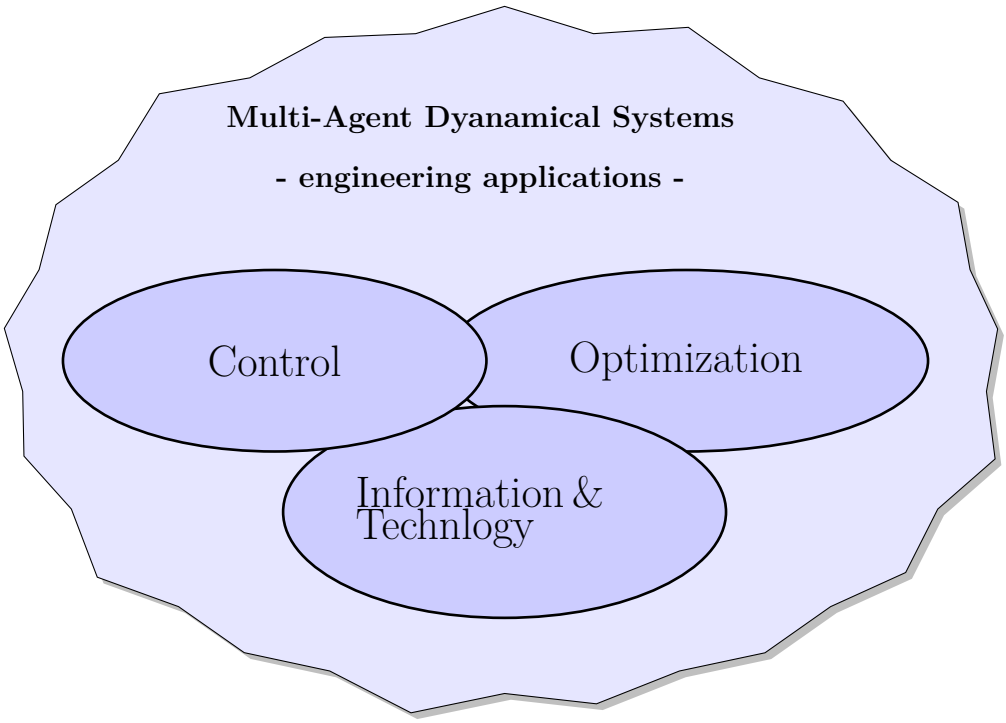


FIGURE 6.1: Multi-Agent dynamical systems.

have considered the more challenging case of agents with dynamics, since we were interested in the class of application relevant to the control community. Needless to say, this means that the control decisions need to take into account not only exogenous factors (e.g., obstacles, reference tracking, etc.) but also the internal (state) dynamics of the agents and their dynamical constraints (as settle-times, nonholonomic characteristics or transmission delays).

Since we have applied set theoretic elements to the multi-agent systems it was natural to consider specific tools in this context. For example, the obstacles and the agents themselves were described as sets (usually convex) and the collision and obstacle avoidance constraints were formulated with set operators like "element inclusion into a set" or "intersection between sets". Secondly, we had to make a series of assumptions without which notions like the "safety region" would have lost their meaning. In particular, we have assumed that perturbations and uncertainties were bounded. Even if sometimes it is difficult to provide a theoretical limit for noises/disturbances affecting a system, in practice it is natural to consider such bounds as well as hard constraints.

The use of the set-theoretic framework was not only a convenient way of describing constraints and behavior, but also provided interesting results on its own. For example, one of the fundamental notions in set theory is the invariance of a set. We have used

this notion to describe invariant safety regions around agents affected by disturbances. Thus we have avoided the need to recalculate the set at each instant of time (as it is usually the case in the literature).

Another important element of our approach, the Mixed-Integer Programming techniques, enabled us to describe in a tractable form the non-convex, non-connected feasible region resulting from typical multi-agent collision and obstacle avoidance constraints. Elements like hyperplane arrangements and cell merging techniques have helped us to describe efficiently such a feasible region. We believe that the results and insights on the description of a non-convex feasible region are useful not only for the multi-agent topic but also in a more general setting of the optimization problems with non-convex constraints.

We have also linked the existence and uniqueness of a tight formation of agents with constraints over the eigenstructure of the state matrices of the agents. This formal analysis has permitted interesting remarks and shown the deep connection between notions like limit cycles, fixed points and formation design and control. It is important to mark that these properties are dependent on the optimization scheme used, but at least, in the centralized (and, in certain conditions, distributed) setting they can be guaranteed.

Lastly, we have applied some of these theoretical results over a challenging practical application. We have evaluated a combination of Model Predictive Control and differential flatness for the flight control of Unmanned Aerial Vehicles (UAVs). In a first stage the MPC flight controllers were fine-tuned and tested in simulation while in a second stage we have executed flight tests with actual UAVs (during a visit at the LSTS laboratory in Porto). We mark that it is not easy to go from designing a controller on the paper to real implementation. Therefore, we believe that the real-time results were extremely promising and a thoroughgoing research in this direction is worthwhile.

6.2 Future developments

All along the manuscript we have discussed various improvements and contributions towards the control of multi-agent dynamical systems under a set-theoretic framework. While hopping we have advanced the state of the art, we are also aware that there is much to be added.

For example, when using set-theoretic tools we have limited mainly to linear time-invariant (LTI) models. This permits relatively easy numerical implementations and covers a large number of situations in control applications. Nonetheless, more complex situations may prove qualitatively different and require new techniques. For example, piecewise affine or switched systems are arguably more able to describe real systems and impose significant modifications to the techniques used in the LTI case. Furthermore, extending the study to the time-delay systems will represent a valuable generalization

by the fact that the networked dimension of the multi-agent systems will allow a natural incorporation in the dynamical model.

In general, there are issues of set theory whose resolution/improvement (even of the off-line design stage procedure) would help in providing better implementations of the control schemes. For example, the computation of RPI sets, reachable sets or convergence towards such a set are still open topics of research in the literature, at least withe respect to their complexity and conservatism. Another limiting factor is the shape of the sets considered. Due to their practicality we have mainly used polyhedral sets in this manuscript but various other choices are available. Taking a step further we can generalize the dynamical models to differential inclusions and use set-valued analysis as design framework. We consider this as a future research direction by use of viability theory [Aubin, 1991]. As briefly detailed in the Introduction chapter of the present manuscript, this framework promises a much more general implementation: the sets are no longer limited to a certain shape and the use of set-valued notions comes naturally (generalizations of the notions of continuity, differentiability or set inclusion).

Solving optimization problems over non-convex regions is not a new issue in the literature and, we have showed that Mixed-Integer Programming (MIP) formulations provide one of the best ways of dealing with this type of problems. With all the valuable improvements that we carried out, the computational complexity is still highly dependent on the MIP formulation and limits its usefulness to relatively small size problems. We believe that we can go further in advancing the novel geometrical interpretation of the non-convex constraints originated from the multi-agent problem. Concentrating on compact ways of describing the feasible region, we will avoid unnecessarily steps towards the mixed-integer formulation. In particular, we want to avoid decomposing explicitly the hyperplane arrangement into cells since this operation carries a significant computational load.

We also believe that there are significant improvements which can be considered in the study of existence and uniqueness for multi-agent formations. In particular we would like to deepen the links between eigenstructure assignment, invariance inducing constraints and limit behavior from the point of view of multi-agent formation. For example in the present manuscript we concentrated on the existence and uniqueness of a fixed point. This may actually not be possible for certain cases (UAV's cannot have a steady position while in flight). In such a case we would like to determine/design (an optimal or economic) limit cycle with all that follows: uniqueness, basin of attraction, stability and so forth.

The experimental results obtained for the real-time flight control of Unmanned Aerial Vehicles (UAVs) were promising and more than that, they have raised challenging theoretical questions. We have obtain a good quality controller but we believe there are still improvements to be considered. We did not took into account the effects of measurement errors or other perturbations and disturbances in the construction of the optimization-based controller. We have made use of the MPC implicit robustness for dealing with

their effect. We note first the need for a better mechanism for wind estimation in order to decrease the uncertainty in the prediction model (e.g., a Kalman filter implementation). Another sensitive point is the discretization and the inter-sample variations: up to now we have assumed a fixed length of the discretization step and computed the input action accordingly. Not in the least, a future improvement of the control mechanism will be the inclusion of the communication delays in the design of the predictive controller. Besides these enhancements, currently undergoing work focuses on the extension to path tracking and the flight formation control which will empower the use of the mixed-integer formulations we have presented in the manuscript.

Appendices

Appendix A

Appendix

A.1 Proof of Proposition 4.1, Section 4.1.2, Chapter 4

a. Consider $x(k) \in X_0$ such that all the future values under LQ dynamics (4.2) reside in X_0. As $(A + BK_{LQ})$ is a Schur matrix it means that for any $\epsilon \in \mathbb{R}_+^*$ there exists a $t_{max} \in \mathbb{N}$ such that $||(A + BK_{LQ})^{t_{max}} x(k)|| < \epsilon$. But, Remark 4.2 states that $0 \notin X_0$, therefore it follows that any $x(k)$ has to transit from X_0 in a finite time. From the definition of X_0 (4.14) it follows that the transit set is X_1, thus concluding the proof.

b. Since inside the polyhedral set X_0 (defined in (4.14)) the result of the optimization problem (4.7) corresponds to the unconstrained optimum (4.2) due to the fact that $h_1^T x(k+1) > 1$, it follows that the polyhedral set X_1 (4.15) corresponds to the constraint activation $h_1^T x(k+1) = 1$. As a consequence, for any $x(k) \in X_1$ one has $x(k+1)$ on the boundary of the feasible region, thus concluding the proof. □

A.2 Proof of Theorem 4.1, Section 4.1.2, Chapter 4

The first-order Karush-Kuhn-Tucker (KKT) optimality conditions [Bemporad et al., 2002] provides the construction of F_1 and G_1. Consider the polyhedral set X_1 described as $X_1 = L_1 \cap Y$, where the set Y defined in (4.13) is invariant with respect to the piecewise affine dynamics (4.16) (the set Y is composed by two regions X_0, X_1, which do not transit outside the set, see Proposition 4.1). Using Lemma A.1 (see the next section) we prove that L_1 defined in (4.11) is a positively invariant set with respect to the dynamics $x(k+1) = (A + BF_1)x(k) + BG_1$ from (4.16). Therefore, taking into account that the set X_1 is the intersection between two invariant sets, results that X_1 is also a positively invariant set.

a. Now, let us consider the case where the fixed point x_e is contained by the invariant set X_1. Since the dynamics $x(k+1) = (A+BF_1)x(k)+BG_1$ associated to X_1 are affine, then the fixed point is:

- *unique* by the fact that over the frontier $X_0 \cap X_1$ the system dynamics correspond to the closed-loop dynamics $x(k+1) = (A+BK_{LQ})x(k)$ which admits a unique fixed point on the origin. But $0 \notin X_1$ and $\{1\} \notin \Lambda(A+BF_1)$, which means that none of the eigenvalues of the closed-loop matrix $A+BF_1$ are equal to 1.
- *stable* by the fact that the dynamics in X_1 contain an eigenvalue at the origin (see Remark 4.3) while the second eigenvalue is inside the unit disc as a consequence of the invariance and continuity.

The local attractivity in X_1 is completed with the attractivity in $Y = X_0 \cup X_1$ as a consequence of Proposition 4.1.

b. Let us consider the point $z(k) \in \mathbb{R}^n$ such that

$$z(k) \in \{x \in \mathbb{R}^2 : h_1^T x(k) = 1\} \cap \{x \in \mathbb{R}^2 : h_c^T x(k) = 1\}, \tag{A.1}$$

with h_c given by (4.12). In order to establish that in the case where the affine dynamics of X_1 is unstable, all the trajectories goes to infinity, it suffices to prove that all future values starting from $z(k)$ as in (A.1) will transit in X_1 on the boundary of the feasible region. We prove this by contradiction.

Assume that $z(k)$ will transit in one step in X_0 (i.e. $z(k+1) \in X_0$). Since by its definition $z(k) \in X_1$ it follows from Proposition 4.1.b that $z(k+1)$ is on the boundary of the feasible region. Consequently we state the following relations:

$$h_1^T z(k) = 1, \tag{A.2}$$

$$h_1^T z(k+1) = h_1^T A_{LQ} z(k) = 1, \tag{A.3}$$

where $A_{LQ} = A + BK_{LQ}$ denotes the closed-loop dynamics of region X_0. Consequently, at the next iteration we have that

$$h_1^T z(k+2) > 1. \tag{A.4}$$

The Cayley-Hamilton theorem states that replacing $A_{LQ} \in \mathbb{R}^{2\times 2}$ in the characteristic polynomial yields to $A_{LQ}^2 + c_1 A_{LQ} + c_2 I = 0$, which is equivalent to

$$A_{LQ}^2 = -c_1 A_{LQ} - c_2 I, \tag{A.5}$$

with $c_1, c_2 \in \mathbb{R}$ and

$$c_1 + c_2 < 1. \tag{A.6}$$

Introducing (A.5) and (A.6) in (A.4) it results that

$$h_1^T z(k+2) = h_1^T(-c_1 A_{LQ} - c_2 I)z(k) = -(c_1 + c_2).$$

We reach a contradiction as long as (A.4) is satisfied. Subsequently, we have shown by contradiction that $z(k+1) \in X_0$ is false, therefore all future values starting from $z(k)$ will transit in X_1.

Let there be V and Λ the Jordan decomposition $V\Lambda V^{-1} = A + BF_1$. Remark 4.3 states that one of the eigenvalues of the state matrix is 0, allowing to write the dynamics associated to the set X_1:

$$x(k+1) = V\Lambda V^{-1}x(k) + BG_1. \tag{A.7}$$

After elementary algebraic operations it results that

$$\begin{bmatrix} y(k+1)_1 \\ y(k+1)_2 \end{bmatrix} = \begin{bmatrix} 0 & \\ & \lambda \end{bmatrix} \begin{bmatrix} y(k)_1 \\ y(k)_2 \end{bmatrix} + \begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix}, \tag{A.8}$$

where $y(k) = \begin{bmatrix} y(k)_1 \\ y(k)_2 \end{bmatrix} = V^{-1}x(k)$, $\begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = V^{-1}BG_1$ and λ is the nonzero eigenvalue of Λ. After k iterations the following relations are obtained:

$$y(k+1)_1 = \delta_1 \tag{A.9}$$

$$y(k+1)_2 = \lambda^k y(0)_2 + \sum_{i=0}^{k-1} \lambda^i \delta_2 \tag{A.10}$$

Using (A.10) we observe that as long as $\lambda \geq 1$ the distance between x_e and $z(k)$ expands at the next iteration, that is

$$||z(k) - x_e|| = |\lambda| \cdot \|z(k+1) - x_e\|. \tag{A.11}$$

In addition, if there exists $V \in \mathbb{R}^{2\times 2}$ such that the conditions (4.17), (4.18) are satisfied, X_1 is an invariant set. Therefore, any point from Y has a trajectory that diverges from x_e ($x_e \notin X_1$). Finally, it results that all the trajectories transit to infinity along the boundary of the feasible region.

A.3 Extension of the set invariance conditions provided in Lemma 2.1, Chapter 2

We use as an instrumental result the following lemma establised in [Bitsoris and Truffet, 2011], which is an adaptation for affine systems of Lemma 2.1. A particular case of this result was provided also in Lemma 4.2, Chapter 4.

Lemma A.1. *Consider the polyhedral set*

$$Z(H,K) = \left\{ x \in \mathbb{R}^n : Hx \leq K \right\},$$

with $(H,K) \in \mathbb{R}^{r\times n} \times \mathbb{R}^r$. *If there exists* $V \in \mathbb{R}^{r\times r}$ *with nonnegative elements such that*

$$H\Phi = VH \quad \textit{and} \tag{A.12}$$
$$(V-I)K + H\Gamma \quad \leq 0, \tag{A.13}$$

then $Z(H,K)$ *is a positively invariant set with respect to the affine dynamics* $x(k+1) = \Phi x(k) + \Gamma$. □

Proof: Suppose $x(k) \in Z$. We want to prove that $x(k+1) \in Z$. By explicitly replacing

$$Hx(k+1) = H(\Phi x(k) + \Gamma) \overset{(A.12)}{=} VHx(k) + H\Gamma,$$

and taking into account the hypothesis that $Hx(k) \leq K$ and V has nonnegative elements it follows that

$$Hx(k+1) \leq VK + H\Gamma.$$

with $VK + H\Gamma = (V-I)K + H\Gamma + K \leq K$ which can then be verified by (A.13). □

Bibliography

W. Achour, H. Piet-Lahanier, and H. Siguerdidjane. Wind field bounded error identification and robust guidance law design for a small-scaled helicopter. *Automatic Control in Aerospace*, 18(1):43–48, 2010.

A.P. Aguiar and J.P. Hespanha. Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. *Automatic Control, IEEE Transactions on*, 52(8):1362–1379, 2007.

A.P. Aguiar, R. Ghabcheloo, A. Pascoal, C. Silvestre, J. Hespanha, and I. Kaminer. Coordinated path-following of multiple underactuated autonomous vehicles with bidirectional communication constraints. In *Proc. International Symposium on Communications, Control and Signal Processing*, 2006.

K.T. Alfriend. Satellite in formation flight. *Encyclopedia of Aerospace Engineering*, 2010.

P. Almeida, R. Bencatel, G. Gonçalves, J.B. Sousa, and C. Ruetz. Experimental results on command and control of unmanned air vehicle systems. In *Proceedings of the 6th IFAC Symposium on Intelligent Autonomous Vehicles*, 2007.

A.N. Andry, E.Y. Shapiro, and J.C. Chung. Eigenstructure assignment for linear systems. *IEEE Transactions on Aerospace and Electronic Systems*, (5):711–729, 1983.

Z. Artstein and S.V. Raković. Feedback and invariance under uncertainty via set-iterates. *Automatica*, 44(2):520–525, 2008.

E. Asarin, O. Bournez, T. Dang, and O. Maler. Approximate reachability analysis of piecewise-linear dynamical systems. *Hybrid Systems: Computation and Control*, pages 20–31, 2000.

J.P. Aubin. *Viability theory*. Birkhauser, 1991.

J.P. Aubin, A. Bayen, and P. Saint-Pierre. *Viability Theory: New Directions*. Springer Verlag, 2011.

D. Avis and K. Fukuda. Reverse search for enumeration. *Discrete Applied Mathematics*, 65(1):21–46, 1996.

T. Balch and R.C. Arkin. Behavior-based formation control for multirobot teams. *Robotics and Automation, IEEE Transactions on*, 14(6):926–939, 1998.

L.E. Barnes, M.A. Fields, and K.P. Valavanis. Swarm formation control utilizing elliptical surfaces and limiting functions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(6):1434–1445, 2009.

T. Basar and G.J. Olsder. *Dynamic noncooperative game theory*, volume 200. SIAM, 1995.

L.D. Baskar, B. De Schutter, and H. Hellendoorn. Decentralized traffic control and management with intelligent vehicles. In *Proceedings of the 9th TRAIL Congress 2006*, 2006.

D. Bauso, L. Giarré, and R. Pesenti. Noncooperative dynamic games for inventory applications: A consensus approach. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 4819–4824, Cancun, Mexico, 9-11 December 2008.

R.W. Beard, J. Lawton, and F.Y. Hadaegh. A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology*, 9(6):777–790, 2001.

R.W. Beard, T.W. McLain, M.A. Goodrich, and E.P. Anderson. Coordinated target assignment and intercept for unmanned air vehicles. *In Proceedings of IEEE Transactions on Robotics and Automation*, 18(6):911–922, 2002.

J. Bellingham, A. Richards, and J.P. How. Receding horizon control of autonomous aerial vehicles. In *Proceedings of the 21th American Control Conference*, volume 5, pages 3741–3746, Anchorage, Alaska, USA, 8-10 May 2002.

A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–428, 1999.

A. Bemporad, F. Borrelli, and M. Morari. Optimal controllers for hybrid systems: Stability and piecewise linear explicit form. In *Proceedings of the 39th IEEE Conference on Decision and Control*, pages 1810–1815, Sydney, NSW, Australia, 12-15 December 2000.

A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

R. Bencatel, M. Faied, J. Sousa, and A.R. Girard. Formation control with collision avoidance. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, pages 591–596, Orlando, Florida, USA, 12-15 December 2011.

L. Bertino and KA Lisæter. The topaz monitoring and prediction system for the atlantic and arctic oceans. *Journal of Operational Oceanography*, 1(2):15–18, 2008.

D.P. Bertsekas. Auction algorithms for network flow problems: A tutorial introduction. *Computational Optimization and Applications*, 1(1):7–66, 1992. ISSN 0926-6003.

D.P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific Belmont, MA, 1995.

D.P. Bertsekas. *Network optimization: continuous and discrete models*. Citeseer, 1998. ISBN 1886529027.

D.P. Bertsekas and D.A. Castanon. Parallel synchronous and asynchronous implementations of the auction algorithm. *Parallel Computing*, 17(6-7):707–732, 1991. ISSN 0167-8191.

R.P. Bitmead, V. Wertz, and M. Gerers. *Adaptive optimal control: the thinking man's GPC*. Prentice Hall Professional Technical Reference, 1991.

G. Bitsoris. On the positive invariance of polyhedral sets for discrete-time systems. *Systems & Control Letters*, 11(3):243–248, 1988.

G. Bitsoris and N. Athanasopoulos. A dual comparison principle for the analysis and design of discrete-time dynamical systems. In *IFAC World Congress*, volume 18, pages 6437–6442, 2011.

G. Bitsoris and L. Truffet. Invariance and monotonicity of nonlinear iterated systems. *Lecture notes in control and information sciences*, 341:407, 2006.

G. Bitsoris and L. Truffet. Positive invariance, monotonicity and comparison of nonlinear systems. *Systems & Control Letters*, 60(12):960–966, 2011.

F. Blanchini. Nonquadratic Lyapunov functions for robust control. *Automatica*, 31(3): 451–461, 1995. ISSN 0005-1098.

F. Blanchini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.

F. Blanchini and S. Miani. *Set-theoretic methods in control*. Birkhauser, 2007.

F. Blanchini and S. Miani. *Set-theoretic methods in control*. Springer, 2008.

L.G. Bleris, P.D. Vouzis, J.G. Garcia, M.G. Arnold, and M.V. Kothare. Pathways for optimization-based drug delivery. *Control Engineering Practice*, 15(10):1280–1291, 2007.

V.D. Blondel, J.M. Hendrickx, and J.N. Tsitsiklis. On krause's multi-agent consensus model with state-dependent connectivity. *IEEE Transactions on Automatic Control*, 54(11):2586–2597, 2009.

H. Bock, M. Diehl, P. Kühl, E. Kostina, J. Schiöder, and L. Wirsching. Numerical methods for efficient and fast nonlinear model predictive control. *Assessment and future directions of nonlinear model predictive control*, pages 163–179, 2007.

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

R.C. Buck. Partition of space. *American Mathematical Monthly*, 50(9):541–544, 1943.

R.E. Burkard. Selected topics on assignment problems. *Discrete Applied Mathematics*, 123(1):257–302, 2002.

B. Burmeister, A. Haddadi, and G. Matylis. Application of multi-agent systems in traffic and transportation. *IEE Proceedings Software Engineering*, 144(1):51–60, 1997.

D.P. Buse, P. Sun, Q.H. Wu, and J. Fitch. Agent-based substation automation. *Power and Energy Magazine*, 1(2):50–55, 2003.

E.F. Camacho and C. Bordons. *Model predictive control*. Springer Verlag, 2004.

E. Camponogara and L.B. De Oliveira. Distributed optimization for model predictive control of linear-dynamic networks. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 39(6):1331–1338, 2009.

N.G. Chetaev. On the instability of equilibrium in some cases where the force function is not maximum. *Prikl. Mat. Mekh*, 16(1), 1952.

D. Chmielewski and V. Manousiouthakis. On constrained infinite-time linear quadratic optimal control. *Systems & Control Letters*, 29(3):121–129, 1996.

P. Colaneri. Dwell time analysis of deterministic and stochastic switched systemsl. In *Proceedings of the 10th IEEE European Control Conference*, pages 15–31, Budapest,Hungary, 23-26 August 2009.

A.W. Colombo, R. Schoop, and R. Neubert. An agent-based intelligent control platform for industrial holonic manufacturing systems. *Industrial Electronics, IEEE Transactions on*, 53(1):322–337, 2006.

L. Consolini, F. Morbidi, D. Prattichizzo, and M. Tosques. Stabilization of a hierarchical formation of unicycle robots with velocity and curvature constraints. *Transactions on Robotics*, 25(5):1176–1184, 2009.

L.B. Cremean, T.B. Foote, J.H. Gillula, G.H. Hines, D. Kogan, K.L. Kriechbaum, J.C. Lamb, J. Leibs, L. Lindzey, C.E. Rasmussen, et al. Alice: An information-rich autonomous vehicle for high-speed desert navigation. *Journal of Field Robotics*, 23(9): 777–810, 2006.

E. Crück and P. Saint-Pierre. Nonlinear impulse target problems under state constraint: A numerical analysis based on viability theory. *Set-Valued Analysis*, 12(4):383–416, 2004.

C.A.R. Crusius and A. Trofino. Sufficient lmi conditions for output feedback control problems. *IEEE TAC*, 44(5):1053–1057, 1999.

J.E. da Silva, B. Terra, R. Martins, and J.B. de Sousa. Modeling and simulation of the lauv autonomous underwater vehicle. In *13th IEEE IFAC International Conference on Methods and Models in Automation and Robotics*, 2007.

M. Daniel and J.C. Daubisse. The numerical problem of using bé zier curves and surfaces in the power basis. *Computer Aided Geometric Design*, 6(2):121–128, 1989.

A.K. Das, R. Fierro, V. Kumar, J.P. Ostrowski, J. Spletzer, and C.J. Taylor. A vision-based formation control framework. *Transactions on Robotics and Automation*, 18(5): 813–825, 2002.

J. De Doná, F. Suryawan, M. Seron, and J. Lévine. A flatness-based iterative method for reference trajectory generation in constrained NMPC. *Int. Workshop on Assesment and Future Direction of Nonlinear Model Predictive Control*, pages 325–333, 2009.

J. Demmel and P. Koev. The accurate and efficient solution of a totally positive generalized vandermonde linear system. *SIAM Journal on Matrix Analysis and Applications*, 27(1):142–152, 2005.

P. Deuflhard, E. Hairer, and J. Zugck. One-step and extrapolation methods for differential-algebraic systems. *Numerische Mathematik*, 51(5):501–516, 1987.

P.S. Dias, J. Pinto, R. Gonçalves, and J.B. Sousa. Enabling a dialog–a c2i infrastructure for unmanned vehicles and sensors. In *In Proceedings of the IEEE International Conference on Autonomous and Intelligent Systems*, pages 1–6, 2010.

M. Diehl, H. Ferreau, and N. Haverbeke. Efficient numerical methods for nonlinear mpc and moving horizon estimation. *Nonlinear Model Predictive Control*, pages 391–417, 2009.

D. V. Dimarogonas, S. G. Loizou, K. J. Kyriakopoulos, and M.M. Zavlanos. A feedback stabilization and collision avoidance scheme for multiple independent non-point agents. *Automatica*, 42(2):229–243, 2006.

D.V. Dimarogonas and K.J. Kyriakopoulos. A feedback control scheme for multiple independent dynamic non-point agents. *International Journal of Control*, 79(12): 1613–1623, 2006.

D.V. Dimarogonas and K.J. Kyriakopoulos. On the rendezvous problem for multiple nonholonomic agents. *IEEE Transactions on Automatic Control*, 52(5):916–922, 2007.

A.L. Dimeas and N.D. Hatziargyriou. Operation of a multiagent system for microgrid control. *Transactions on Power Systems*, 20(3):1447–1455, 2005.

W.B. Dunbar and R.M. Murray. Model predictive control of coordinated multi-vehicle formations. In *Proceedings of the 41th IEEE Conference on Decision and Control*, volume 4, pages 4631–4636, Las Vegas, Nevada, USA, 10-13 December 2002.

W.B. Dunbar and R.M. Murray. Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42(4):549–558, 2006.

M.G. Earl and R. D'Andrea. Modeling and control of a multi-agent system using mixed integer linear programming. In *Proceedings of the 40th IEEE Conference on Decision and Control*, volume 1, pages 107–111, Orlando, Florida, USA, 4-7 December 2001.

M.G. Earl and R. D'Andrea. Iterative MILP methods for vehicle-control problems. *IEEE Transactions on Robotics*, 21(6):1158?1167, 2005.

L. Fagiano, M. Canale, and M. Milanese. Set membership approximation of discontinuous nmpc laws. In *Proceedings of Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, pages 8636–8641, Shanghai, P.R. China, 16-18 December 2009.

P. Falcone, M. Tufo, F. Borrelli, J. Asgari, and H.E. Tseng. A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems. In *Decision and Control, 2007 46th IEEE Conference on*, pages 2980–2985. IEEE, 2007.

Z. Fang, W. Song, J. Zhang, and H. Wu. Experiment and modeling of exit-selecting behaviors during a building evacuation. *Statistical Mechanics and its Applications*, 389(4):815–824, 2010.

G.E. Farin. *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide*. Morgan Kaufmann, 5th edition, 2001.

R.T. Farouki and V.T. Rajan. On the numerical condition of polynomials in bernstein form. *Computer Aided Geometric Design*, 4(3):191–216, 1987.

J.A. Fax and R.M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

J.A. Ferrez and K. Fukuda. Implementations of lp-based reverse search algorithms for the zonotope construction and the fixed-rank convex quadratic maximization in binary variables using the zram and the cddlib libraries. Technical report, Mcgill, 2002.

R. Fierro, A.K. Das, V. Kumar, and J.P. Ostrowski. Hybrid control of formations of robots. In *Proceedings the IEEE International Conference on Robotics and Automation*, volume 1, pages 157–162, 2001.

M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Flatness and defect of non-linear systems: introductory theory and examples. *International journal of control*, 61(6): 1327–1361, 1995.

F. Fontes, D. Fontes, and A. Caldeira. Model predictive control of vehicle formations. *Optimization and Cooperative Control Strategies*, pages 371–384, 2009.

T.I. Fossen. Guidance and control of ocean vehicles. *New York*, 1994.

K. Fukuda. From the zonotope construction to the Minkowski addition of convex polytopes. *Journal of Symbolic Computation*, 38(4):1261–1272, 2004.

Y. Gao, J. Lygeros, M. Quincampoix, and N. Seube. On the control of uncertain impulsive systems: Approximate stabilization and controlled invariance. *International Journal of Control*, 77(16):1393–1407, 2004.

C.E. Garcia, D.M. Prett, and M. Morari. Model predictive control: theory and practice - a survey. *Automatica*, 25(3):335–348, 1989.

M.R. Garey and D.S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness. A Series of Books in the Mathematical Sciences*. WH Freeman and Company, San Francisco, Calif, 1979.

T. Geyer, F.D. Torrisi, and M. Morari. Optimal complexity reduction of piecewise affine models based on hyperplane arrangements. In *Proceedings of the 23th American Control Conference*, volume 2, pages 1190–1195, Boston, Massachusetts, USA, 30 June - 2 July 2004.

T. Geyer, F.D. Torrisi, and M. Morari. Optimal complexity reduction of polyhedral piecewise affine systems. *Automatica*, 44(7):1728–1740, 2008. ISSN 0005-1098.

E.G. Gilbert and K.T. Tan. Linear systems with state and control constraints: The theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36(9):1008–1020, 1991.

A. Girard, C. Le Guernic, and O. Maler. Efficient computation of reachable sets of linear time-invariant systems with inputs. *Hybrid Systems: Computation and Control*, pages 257–271, 2006.

A.R. Girard, J.B. de Sousa, and J.K. Hedrick. Coordinated Control of Agent Formations in Uncertain, Dynamic Environments. In *Proceedings of the European Control Conference, Porto, Portugal*, 2001.

A.R. Girard, J.B. De Sousa, and J.K. Hedrick. A selection of recent advances in networked multivehicle systems. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 219(1):1–14, 2005.

R. Gonçalves, S. Ferreira, J. Pinto, J. B. Sousa, and G. Gonçalves. Authority sharing in mixed initiative control of multiple uninhabited aerial vehicles. *Engineering Psychology and Cognitive Ergonomics*, pages 530–539, 2011.

G.C. Goodwin, M.M. Seron, and J.A. De Dona. *Constrained control and estimation: an optimisation approach*. Springer Verlag, 2005.

G.C. Goodwin, M.M. Seron, R.H. Middleton, M. Zhang, B.F. Hennessy, P.M. Stone, and M. Menabde. Receding horizon control applied to optimal mine planning. *Automatica*, 42(8):1337–1342, 2006.

G.C. Goodwin, M.M. Seron, and D.Q. Mayne. Optimization opportunities in mining, metal and mineral processing. *Annual reviews in control*, 32(1):17–32, 2008.

C. Grant and C. Shaw. Operational oceanographic needs for the offshore oil and gas industry. *GOOS data products and services bulletin. Volume*, 1, 2001.

D. Grundel, R. Murphey, and P.M. Pardalos. *Cooperative systems, Control and optimization*, volume 588. Springer Verlag, 2007.

H. Haimovich, E. Kofman, and M.M. Seron. Systematic ultimate bound computation for sampled-data systems with quantization. *Automatica*, 43(6):1117–1123, 2007.

H. Haimovich, E. Kofman, M.M. Seron, I. y Agrimensura, and R. de Rosario. Analysis and improvements of a systematic componentwise ultimate-bound computation method. In *17th IFAC world congress, Seoul, South Korea*, 2008.

O. Hallefjord Kurt. Solving large scale generalized assignment problems–An aggregation/disaggregation approach. *European Journal of Operational Research*, 64(1):103–114, 1993.

Y. Hao. *A practical framework for formation planning and control of multiple unmanned ground vehicles*. PhD thesis, University of Delaware, 2004.

Y. Hao and S.K. Agrawal. Formation planning and control of ugvs with trailers. *Autonomous Robots*, 19(3):257–270, 2005.

W.P.M.H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.

A. Hegyi, B. De Schutter, and J. Hellendoorn. Optimal coordination of variable speed limits to suppress shock waves. *Intelligent Transportation Systems, IEEE Transactions on*, 6(1):102–112, 2005.

B. Houska, H.J. Ferreau, and M. Diehl. Acado toolkit: An open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

M. Hovd and S. Olaru. Piecewise quadratic lyapunov functions for stability verification of approximate explicit mpc. *Modeling, Identification and Control*, 31(2):45–53, 2010.

J. How, E. King, and Y. Kuwata. Flight demonstrations of cooperative control for uav teams. In *AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit*, pages 20–23, 2004.

A. Jadbabaie, J. Lin, and A.S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6): 988–1001, 2003. ISSN 0018-9286.

M. Ji and M. Egerstedt. Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics*, 23(4):693–703, 2007.

M. Jünger, T.M. Liebling, D. Naddef, G. Nemhauser, and W.R. Pulleyblank. *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*. Springer Verlag, 2009.

T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.

T. Keviczky and G.J. Balas. Software-enabled receding horizon control for autonomous unmanned aerial vehicle guidance. *Journal of Guidance Control and Dynamics*, 29(3): 680–694, 2006.

T. Keviczky, F. Borrelli, and G.J. Balas. Decentralized receding horizon control for large scale dynamically decoupled systems. *Automatica*, 42(12):2105–2115, 2006.

O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90, 1986. ISSN 0278-3649.

R. Khosla and T.S. Dillon. *Engineering intelligent hybrid multi-agent systems*. Springer, 1997.

H.J. Kim, D.H. Shim, and S. Sastry. Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles. In *Proceedings of the 21th American Control Conference*, volume 5, pages 3576–3581, Anchorage, Alaska, USA, 8-10 May 2002.

K. Kobayashi and J. Imura. Modeling of discrete dynamics for computational time reduction of Model Predictive Control. In *Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems*, pages 628–633, Kyoto, Japan, 24-28 July 2006.

D. E. Koditschek. Task encoding: Toward a scientific paradigm for robot planning and control. *Robotics and autonomous systems*, 9(1):5–39, 1992.

E. Kofman, H. Haimovich, and M.M. Seron. A systematic method to obtain ultimate bounds for perturbed systems. *International Journal of Control*, 80(2):167–178, 2007a.

E. Kofman, M.M. Seron, and H. Haimovich. Robust Control Design with Guaranteed State Ultimate Bound. In *Proceedings of the 3rd International Conference on Integrated Modeling and Analysis in Applied Control and Automation*, Buenos Aires, Argentina, 8-10 February 2007b.

E. Kofman, M.M. Seron, and H. Haimovich. Control design with guaranteed ultimate bound for perturbed systems. *Automatica*, 44(7):1815–1821, 2008.

K.I. Kouramas, S.V. Raković, E.C. Kerrigan, J.C. Allwright, and D.Q. Mayne. On the minimal robust positively invariant set for linear difference inclusions. In *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference*, pages 2296–2301, Seville, Spain, 12-15 December 2005.

H.W. Kuhn. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

A.B. Kurzhanskiĭ and I. Vályi. *Ellipsoidal calculus for estimation and control*. Iiasa Research Center, 1997.

G. Lafferriere, J. Caughman, and A. Williams. Graph theoretic methods in the stability of vehicle formations. In *Proceedings of the 23th American Control Conference*, pages 3729–3734, Boston, Massachusetts, USA, 30 June - 2 July 2004.

V.B. Larin. About the inverse problem of optimal control. *Journal of Applied and Computational Mathematics*, 2(2):90–97, 2003.

J. Lévine. *Analysis and control of nonlinear systems: A flatness-based approach*. Springer Verlag, 2009.

S. Li, Y. Zhang, and Q. Zhu. Nash-optimization enhanced distributed model predictive control applied to the shell benchmark problem. *Information Sciences*, 170(2):329–349, 2005.

Z. Li and J. Canny. *Nonholonomic motion planning*, volume 192. Kluwer Academic Pub, 1993.

Z. Lin, B. Francis, and M. Maggiore. Necessary and sufficient graphical conditions for formation control of unicycles. *Automatic Control, IEEE Transactions on*, 50(1): 121–127, 2005.

V. Loechner and D.K. Wilde. Parameterized polyhedra and their vertices. *International Journal of Parallel Programming*, 25(6):525–549, 1997.

K. Loskot, A. Polanski, and R. Rudnicki. Further comments on vector norms as lyapunov functions for linear systems. *Automatic Control, IEEE Transactions on*, 43(2):289–291, 1998.

J. Marden and J. S. Shamma. Autonomous vehicle target assignment: a game theoretical formulation. *ASME Journal of Dynamic Systems, Measurement, and Control*, pages 584–596, 2007.

V. Marik and D. McFarlane. Industrial adoption of agent-based technologies. *Intelligent Systems, IEEE*, 20(1):27–35, 2005.

G.L. Mariottini, F. Morbidi, D. Prattichizzo, N. Vander Valk, N. Michael, G. Pappas, and K. Daniilidis. Vision-based localization for leader–follower formation control. *Transactions on Robotics and Automation*, 25(6):1431–1438, 2009.

R. Martins, P.S. Dias, E.R.B. Marques, J. Pinto, J.B. Sousa, and F.L. Pereira. Imc: A communication protocol for networked vehicles and sensors. In *OCEANS 2009-EUROPE*, pages 1–6. IEEE, 2009.

D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

D.Q. Mayne, M.M. Seron, and S.V. Raković. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41(2):219–224, 2005.

S.D.J. McArthur, E.M. Davidson, V.M. Catterson, A.L. Dimeas, N.D. Hatziargyriou, F. Ponci, and T. Funabashi. Multi-agent systems for power engineering applications - part ii: technologies, standards, and tools for building multi-agent systems. *IEEE Transactions on Power Systems*, 22(4):1753–1759, 2007.

M. Mesbahi and M. Egerstedt. *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

M.B. Milam, R. Franz, and R.M. Murray. Real-time constrained trajectory generation applied to a flight control experiment. In *Proceedings of the 15th IFAC World Congress*, Barcelona, Spain, 21-26 July 2002.

M. Morari, J.H. Lee, C. Garcia, and DM Prett. Model predictive control. *Preprint*, 2002.

T.S. Motzkin, H. Raiffa, G.L. Thompson, and R.M. Thrall. The double description method. *Contributions to the theory of games*, 2:51, 1959.

R. Murphey and P. M. Pardalos. *Cooperative control and optimization*, volume 66. Springer, 2002.

R.M. Murray. Nonlinear control of mechanical systems: A lagrangian perspective. *Annual Reviews in Control*, 21:31–42, 1997.

R.M. Murray. Optimization-based control. *Technical Report, California Institute of Technology, CA*, 2009.

R.M. Murray, J.W. Burdick, P. Perona, L.B. Cremean, K. Kriechbaum, S. Pfister, T. Foote, J. Gillula, J. Lamb, and A. Stewart. Darpa technical paper: Team caltech. Technical report, Retrieved 10/25/2005 from http://www. darpa. mil/grandchallenge/TechPapers. html, 2005.

R.R. Negenborn, B. De Schutter, and J. Hellendoorn. Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. *Engineering Applications of Artificial Intelligence*, 21(3):353–366, 2008.

R.R. Negenborn, P.J. van Overloop, T. Keviczky, and B. De Schutter. Distributed model predictive control of irrigation canals. *Networks and Heterogeneous Media (NHM)*, 4 (2):359–380, 2009.

Y. Nesterov and A. Nemirovsky. Interior point polynomial methods in convex programming. *Studies in applied mathematics*, 13:1993, 1994.

J.S. Norris, M.W. Powell, M.A. Vona, P.G. Backes, and J.V. Wick. Mars exploration rover operations with the science activity planner. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4618–4623, 2005.

C. Ocampo-Martinez and V. Puig. Fault-tolerant model predictive control within the hybrid systems framework: Application to sewer networks. *International Journal of Adaptive Control and Signal Processing*, 23(8):757–787, 2009.

P. Ogren, M. Egerstedt, and X. Hu. A control lyapunov function approach to multi-agent coordination. In *Proceedings of the 40th IEEE Conference on Decision and Control*, volume 2, pages 1150–1155, Orlando, Florida, USA, 4-7 December 2001.

S. Olaru and D. Dumur. A parameterized polyhedra approach for explicit constrained predictive control. In *Proceedings of the 43th IEEE Conference on Decision and Control*, Paradise Island, Bahamas, 14-17 December 2004.

S. Olaru and D. Dumur. Avoiding constraints redundancy in predictive control optimization routines. *IEEE Transactions on Automatic Control*, 50(9):1459–1465, 2005.

S. Olaru, J.A. De Dona, and M.M. Seron. Positive invariant sets for fault tolerant multisensor control schemes. In *Proc. of the 17th IFAC World Congress*, pages 1224–1229, Seoul, South Korea, 6-11 July 2008.

S. Olaru, D. Dumur, and S. Dobre. On the geometry of predictive control with nonlinear constraints. *Informatics in Control, Automation and Robotics*, pages 301–314, 2009.

R. Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, 2006.

P. Orlik. Hyperplane arrangements. In *Encyclopedia of Optimization*, pages 1545–1547. Springer US, 2009. ISBN 978-0-387-74759-0.

P. Orlik and H. Terao. *Arrangements of hyperplanes*, volume 300. Springer, 1992.

A.J. Osiadacz. Integer and combinatorial optimization, George L. Nemhauser and Laurence A. Wolsey, Wiley-Interscience Series in Discrete Mathematics and Optimization, New York, 1988. *International Journal of Adaptive Control and Signal Processing*, 4 (4):333–344, 1990.

K.Z. Østergaard, P. Brath, and J. Stoustrup. Estimation of effective wind speed. In *Journal of Physics: Conference Series*, volume 75, pages 12–82. IOP Publishing, 2007.

J. Ousingsawat and M.E. Campbell. Establishing trajectories for multi-vehicle reconnaissance. In *Proceedings of the 22nd AIAA Guidance, Navigation, and Control Conference*, pages 2188–2199, Providence, Rhode Island, USA, 16-19 August 2004.

P.J. Overloop, R.R. Negenborn, B.D. Schutter, and N.C. Giesen. Predictive Control for National Water Flow Optimization in The Netherlands. *Intelligent Infrastructures*, pages 439–461, 2010.

U.S. Palekar, M.H. Karwan, and S. Zionts. A branch-and-bound method for the fixed charge transportation problem. *Management Science*, 36(9):1092–1105, 1990.

G. Pannocchia, S.J. Wright, and J.B. Rawlings. Existence and computation of infinite horizon model predictive control with active steady-state input constraints. *IEEE TAC*, 48(6):1002–1006, 2003. ISSN 0018-9286.

P. M. Pardalos. Hyperplane arrangements in optimization. In *Encyclopedia of Optimization*, pages 1547–1548. Springer US, 2009. ISBN 978-0-387-74759-0.

R.S. Parker, F.J. Doyle, et al. Control-relevant modeling in drug delivery. *Advanced drug delivery reviews*, 48(2-3):211–228, 2001.

R.B. Patel and P.J. Goulart. Trajectory generation for aircraft avoidance maneuvers using online optimization. *Journal of guidance, control, and dynamics*, 34(1):218–230, 2011.

J. Pavón, J. Gómez-Sanz, A. Fernández-Caballero, and J.J. Valencia-Jiménez. Development of intelligent multisensor surveillance systems with agents. *Robotics and Autonomous Systems*, 55(12):892–903, 2007.

J. Pinto, P. Calado, J. Braga, P. Dias, R. Martins, and E. Marques. Implementation of a control arhitecture for networked vehicle systems. In *Proceedings of the IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles*, 2012.

E.N. Pistikopoulos, M.C. Georgiadis, and V. Dua. *Multi-parametric programming*. Wiley-vch, 2007.

I. Prodan, S. Olaru, S. Stoica, and S.-I Niculescu. Path following with collision avoidance and velocity constraints for multi-agent group formations. *Annals of the University of Craiova, Series: Automation, Computers, Electronics and Mechatronics*, 7(34):33–38, 2010. doi: ISSN:1841-0626.

I. Prodan, S. Olaru, C. Stoica, and S.-I. Niculescu. On the limit behavior of multi-agent systems. In *The 8th IEEE International Conference on Informatics in Control, Automation and Robotics*, pages 344–349, Noordwijkerhout, The Netherlands, 28-31 July 2011a.

I. Prodan, S. Olaru, C. Stoica, and S.-I. Niculescu. Predictive control for tight group formation of multi-agent systems. In *In Proceedings of the 18th IFAC World Congress*, pages 138–143, Milan, Italy, 28 August - 2 September 2011b.

I. Prodan, R. Bencatel, S. Olaru, J. Sousa, C. Stoica, and S.-I. Niculescu. Predictive control for autonomous aerial vehicles trajectory tracking. In *In Proceedings of the IFAC Nonlinear Model Predictive Control Conference*, pages 508–513, Noordwijkerhout, The Netherlands, 23-27 August 2012a. available upon request.

I. Prodan, G. Bitsoris, S. Olaru, C. Stoica, and S.-I. Niculescu. On the limit behavior for multi-agent dynamical systems. In *The IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles*, Porto, Portugal, 10-12 April 2012b.

I. Prodan, S. Olaru, C. Stoica, and S.-I. Niculescu. On the tight formation for multi-agent dynamical systems. In *Agents and Multi-agent Systems Technologies and Applications*, volume LNAI 7372, pages 554–565. Springer, 2012c.

I. Prodan, S. Olaru, C. Stoica, and S.-I. Niculescu. Predictive control for trajectory tracking and decentralized navigation of multi-agent formations. In *The 4th IEEE International Conference on Agents and Artificial Intelligence*, pages 209–214, Vilamoura, Portugal, 6-8 February 2012d.

I. Prodan, F. Stoican, S. Olaru, and S.-I. Niculescu. Enhancements on the Hyperplanes Arrangements in Mixed-Integer Techniques. *Journal of Optimization Theory and Applications*, 154(2):549–572, 2012e. doi: 10.1007/s10957-012-0022-9.

I. Prodan, F. Stoican, S. Olaru, C. Stoica, and S.-I. Niculescu. Mixed-integer programming techniques in distributed mpc problems. In *Distributed MPC made easy*, volume 1, pages 273 – 288. Springer, 2012f. to appear in 2013.

S.J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control engineering practice*, 11(7):733–764, 2003.

Z. Qu. *Cooperative control of dynamical systems: applications to autonomous vehicles.* Springer, 2009.

S.V. Raković. Minkowski algebra and Banach Contraction Principle in set invariance for linear discrete time systems. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 2169–2174, New Orleans, Louisiana, USA, 12-14 December 2007.

S.V. Raković and D.Q. Mayne. Robust model predictive control for obstacle avoidance: discrete time case. *Assessment and Future Directions of Nonlinear Model Predictive Control*, pages 617–627, 2007.

S.V. Raković, E.C. Kerrigan, K.I. Kouramas, and D.Q. Mayne. Invariant approximations of the minimal robust positively invariant set. *IEEE Transactions on Automatic Control*, 50(3):406–410, 2005.

S.V. Rakovic, F. Blanchini, E. Cruck, and M. Morari. Robust obstacle avoidance for constrained linear discrete time systems: A set-theoretic approach. In *Decision and Control, 2007 46th IEEE Conference on*, pages 188–193. IEEE, 2007.

S.V. Rakovic, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen. Fully parameterized tube mpc. In *Proceedings of the 18th IFAC World Congress*, pages 197–202, Milano, Italy, 28 August-2 September 2011.

J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design*. Madison, WI: Nob Hill Publishing, LCC, 2009.

W. Ren and R.W. Beard. Decentralized scheme for spacecraft formation flying via the virtual structure approach. *Journal of Guidance, Control, and Dynamics*, 27(1):73–82, 2004.

M. Reyhanoglu, A. van der Schaft, N.H. McClamroch, and I. Kolmanovsky. Dynamics and control of a class of underactuated mechanical systems. *Automatic Control, IEEE Transactions on*, 44(9):1663–1671, 1999.

A. Richards and J.P. How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *Proceedings of the 21th American Control Conference*, pages 1936–1941, Anchorage, Alaska, USA, 8-10 May 2002.

A. Richards and J.P. How. Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility. In *Proceedings of the 24th American Control Conference*, volume 5, pages 4034–4040, Portland, Oregon, USA, 8-10 June 2005.

A. Richards, J. Bellingham, M. Tillerson, and J. How. Coordination and control of multiple uavs. *In AIAA Guidance, Navigation, and Control Conference*, Monterey, CA, 2002.

E. Rimon and D.E. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on robotics and automation*, 8(5):501–518, 1992. ISSN 1042-296X.

J.A. Rossiter. *Model-based predictive control: a practical approach*. CRC, 2003.

P. Rouchon, P. Martin, and R.M. Murray. Flat systems, equivalence and trajectory generation. Technical report, Tech. Rep. CDS 2003-008, California Institute of Technology, Pasadena, Calif, USA, 2003.

R. Rudell and A. Sangiovanni-Vincentelli. Espresso-mv: Algorithms for multiple-valued logic minimization. In *Proc. IEEE Custom Integrated Circuits Conf*, pages 230–234, 1985.

R.O. Saber, W.B. Dunbar, and R.M. Murray. Cooperative control of multi-vehicle systems using cost graphs and optimization. In *Proceedings of the 22th American Control Conference*, volume 3, pages 2217–2222, Denver, Colorado, USA, 4-6 June 2003.

R. Scattolini. Architectures for distributed and hierarchical model predictive control-a review. *Journal of Process Control*, 19(5):723–731, 2009.

D.P. Scharf, F.Y. Hadaegh, and S.R. Ploen. A survey of spacecraft formation flying guidance and control. part ii: control. In *Proceedings of the 23th American Control Conference*, volume 4, pages 2976–2985, Boston, Massachusetts, USA, 30 June - 2 July 2004.

H. Schaub, S.R. Vadali, J.L. Junkins, and K.T. Alfriend. Spacecraft formation flying control using mean orbit elements. *Astrodynamics 1999*, pages 163–181, 2000.

J. Schneider. Pathway Toward a Mid-Infrared Interferometer for the Direct Characterization of Exoplanets. *Arxiv preprint arXiv:0906.0068*, 2009.

R. Schneider. *Convex bodies: the Brunn-Minkowski theory*, volume 44. Cambridge Univ Pr, 1993.

T. Schouwenaars, B. De Moor, E. Feron, and J. How. Mixed integer programming for multi-vehicle path planning. In *Proceedings of the 2nd IEEE European Control Conference*, pages 2603–2608, Porto, Portugal, 4-7 September 2001. Citeseer.

T. Schouwenaars, M. Valenti, E. Feron, and J. How. Implementation and flight test results of milp-based uav guidance. In *Aerospace Conference*, pages 1–13. IEEE, 2005.

C. Schumacher, P. R. Chandler, and S. R. Rasmussen. Task allocation for wide area search munitions. In *Proceedings of the 21th American Control Conference*, volume 3, pages 1917–1922, Anchorage, Alaska, USA, 8-10 May 2002.

C. Schumacher, P. Chandler, and M. Pachter. UAV task assignment with timing constraints. *In AIAA Guidance, Navigation, and Control Conference*, Austin, Texas, 2003.

L.L. Schumaker. *Spline functions: basic theory*. Cambridge Univ Pr, 2007.

M.M. Seron, J.A. De Dona, and G.C. Goodwin. Global analytical model predictive control with input constraints. In *Proceedings of the 39th IEEE Conference on Decision and Control*, volume 1, pages 154–159, Sydney, NSW, Australia, 12-15 December 2000.

M.M. Seron, X.W. Zhuo, J.A. De Doná, and J.J. Martínez. Multisensor switching control strategy with fault tolerance guarantees. *Automatica*, 44(1):88–97, 2008.

N.I. Shaikh and V. Prabhu. Model predictive controller for cryogenic tunnel freezers. *Journal of Food Engineering*, 80(2):711–718, 2007.

J.S. Shamma. *Cooperative control of distributed multi-agent systems*. Wiley Online Library, 2007.

W. Shen, Q. Hao, H.J. Yoon, and D.H. Norrie. Applications of agent-based systems in intelligent manufacturing: An updated review. *Advanced Engineering Informatics*, 20 (4):415–431, 2006.

H. Sira-Ramírez and S. Agrawal. *Differential Flatness*. Marcel Dekker, New York, 2004.

J.M. Soares, A.P. Aguiar, A.M. Pascoal, and M. Gallieri. Triangular formation control using range measurements: An application to marine robotic vehicles. In *Proceedings of the IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles*, 2012.

E. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control*, 26(2):346–358, 1981.

J. Sousa, A.R. Girard, J.K. Hedrick, and F. Kretz. Real-time hybrid control of mobile offshore base scaled models. In *Proceedings of the 19th American Control Conference*, pages 682–686, Chicago,Illinois, USA, 28-30 June 2000.

J.B. Sousa, T. Simsek, and P. Varaiya. Task planning and execution for uav teams. In *Proceedings of the 43th IEEE Conference on Decision and Control*, volume 4, pages 3804–3810, Paradise Island, Bahamas, 14-17 December 2004.

S. Srinathkumar and R.P. Rhoten. Eigenvalue/eigenvector assignment for multivariable systems. *Electronics Letters*, 11(6):124–125, 1975.

D. Srinivasan. *Innovations in Multi-Agent Systems and Application–1*, volume 1. Springer, 2010.

F. Stoican. *Fault tolerant control based on set-theoretic methods*. PhD thesis, Supélec, 2011.

F. Stoican, S. Olaru, J.A. De Doná, and M.M. Seron. Zonotopic ultimate bounds for linear systems with bounded disturbances. In *Proceedings of the 18th IFAC World Congress*, pages 9224–9229, Milano, Italy, 28 August-2 September 2011a.

F. Stoican, I. Prodan, and S. Olaru. On the hyperplanes arrangements in mixed-integer techniques. In *Proceedings of the 30th American Control Conference*, pages 1898–1903, San Francisco, California, USA, 29 June-1 July 2011b.

F. Stoican, I. Prodan, and S. Olaru. Enhancements on the hyperplane arrangements in mixed integer techniques. In *In Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, pages 3986–3991, Orlando, Florida, USA, 12-15 December 2011c.

F. Stoican, S. Olaru, M.M. Seron, and J.A. De Doná. Reference governor design for tracking problems with fault detection guarantees. *Journal of Process Control*, 22(5): 829–836, 2012. doi: 10.1016/j.jprocont.2012.02.004.

F. Suryawan. *Constrained Trajectory Generation and Fault Tolerant Control Based on Differential Flatness and B-splines*. PhD thesis, School of Electrical Engineering and Computer Science, The University of Newcastle, Australia, 2012.

F. Suryawan, J. De Dona, and M. Seron. Methods for trajectory generation in a magnetic-levitation system under constraints. In *18th Mediterranean Conference on Control and Automation*, pages 945–950, 2010.

K.H. Tan and M.A. Lewis. Virtual structures for high-precision cooperative mobile robotic control. In *In Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, volume 1, pages 132–139, 1996.

H. Tanner, A. Jadbabaie, and G. Pappas. Flocking in teams of nonholonomic agents. *Cooperative Control*, pages 458–460, 2005.

H.G. Tanner, A. Jadbabaie, and G.J. Pappas. Flocking in fixed and switching networks. *IEEE Transactions on Automatic Control*, 52(5):863–868, 2007. ISSN 0018-9286.

M. Tillerson and J.P. How. Advanced guidance algorithms for spacecraft formation-keeping. In *Proceedings of the 21th American Control Conference*, volume 4, pages 2830–2835, Anchorage, Alaska, USA, 8-10 May 2002.

A. Ulbig, S. Olaru, D. Dumur, and P. Boucher. Explicit nonlinear predictive control for a magnetic levitation system. *Asian Journal of Control*, 12(3):434–442, 2010.

B. Vaglienti and M. Niculescu. Hardware in the loop simulator for the piccolo avionics. *Cloud Cap Technology (www.cloudcaptech.com)*, 2004.

B. Vaglienti, M. Niculescu, J. Becker, and D. Miley. Piccolo system user's guide. *Cloud Cap Technology (www.cloudcaptech.com)*, October 12 2011.

K. Valavanis. *Advances in unmanned aerial vehicles: state of the art and the road to autonomy*, volume 33. Springer Verlag, 2007.

M.J. Van Nieuwstadt and R.M. Murray. Real-time trajectory generation for differentially flat systems. *International Journal of Robust and Nonlinear Control*, 8(11):995–1020, 1998.

M. Vassilaki, J.C. Hennet, and G. Bitsoris. Feedback control of linear discrete-time systems under state and control constraints. *International Journal of Control*, 47(6): 1727–1735, 1988.

A.N. Venkat, J.B. Rawlings, and S.J. Wright. Stability and optimality of distributed model predictive control. In *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference*, pages 6680–6685, Seville, Spain, 12-15 December 2005.

J.P. Vielma and G.L. Nemhauser. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Mathematical Programming*, 128(1):49–72, 2011.

M.P. Vitus, V. Pradeep, G. Hoffmann, S.L. Waslander, and C.J. Tomlin. Tunnel-milp: Path planning with sequential convex polytopes. In *AIAA Guidance, Navigation, and Control Conference*, Honolulu, Hawaii, USA, 18-21 August 2008.

S. Wiggins. *Global bifurcations and chaos: analytical methods*, volume 73. Springer-Verlag New York, 1988.

J.H. Wilkinson. *The algebraic eigenvalue problem*, volume 155. Oxford Univ. Press, 1965.

M. Wooldridge. *An introduction to multiagent systems*. Wiley & Sons, New York, USA, 2002.

M. Wooldridge, N.R. Jennings, et al. Intelligent agents: Theory and practice. *Knowledge engineering review*, 10(2):115–152, 1995.

M. Wooldridge, J. Müller, and M. Tambe. *Intelligent Agents II-Agent Theories, Architectures, and Languages*, volume 1037. Springer, 1997.

Q.D. Wu, D. Xue, and J. Yao. Consensus analysis of networked multi-agent systems. *Physics Procedia*, 3(5):1921–1931, 2010.

F. Yamaguchi. *Curves and surfaces in computer aided geometric design*. Springer-Verlag Berlin, 1988.

T. Zaslavsky. Counting the faces of cut-up spaces. *American Mathematical Society*, 81 (5), 1975.

D.Z. Zhang, A.I. Anosike, M.K. Lim, and O.M. Akanle. An agent-based approach for e-manufacturing and supply chain integration. *Computers & Industrial Engineering*, 51(2):343–360, 2006.

G.M. Ziegler. *Lectures on polytopes*. Springer, 1995.

Resumé

L'objectif de cette thèse est de proposer des solutions aux problèmes liés à la commande optimale de systèmes dynamiques multi-agents en présence de contraintes. Des éléments de la théorie de commande et d'optimisation sont appliqués à différents problèmes impliquant des formations de systèmes multi-agents. La thèse examine le cas d'agents soumis à des contraintes dynamiques. Pour faire face à ces problèmes, les concepts bien établis tels que la théorie des ensembles, la platitude différentielle, la commande prédictive (Model Predictive Control - MPC), la programmation mixte en nombres entiers (Mixed-Integer Programming - MIP) sont adaptés et améliorés. En utilisant ces notions théoriques, ce travail de thèse a porté sur les propriétés géométriques de la formation d'un groupe multi-agents et propose un cadre de synthèse original qui exploite cette structure. En particulier, le problème de conception de formation et les conditions d'évitement des collisions sont formulés comme des problèmes géométriques et d'optimisation pour lesquels il existe des procédures de résolution. En outre, des progrès considérables dans ce sens ont été obtenus en utilisant de façon efficace les techniques MIP (dans le but d'en déduire une description efficace des propriétés de non convexité et de non connexion d'une région de faisabilité résultant d'une collision de type multi-agents avec des contraintes d'évitement d'obstacles) et des propriétés de stabilité (afin d'analyser l'unicité et l'existence de configurations de formation de systèmes multi-agents). Enfin, certains résultats théoriques obtenus ont été appliqués dans un cas pratique très intéressant. On utilise une nouvelle combinaison de la commande prédictive et de platitude différentielle (pour la génération de référence) dans la commande et la navigation de véhicules aériens sans pilote (UAVs).

Abstract

The goal of this thesis is to propose solutions for the optimal control of multi-agent dynamical systems under constraints. Elements from control theory and optimization are merged together in order to provide useful tools which are further applied to different problems involving multi-agent formations. The thesis considers the challenging case of agents subject to dynamical constraints. To deal with these issues, well established concepts like set-theory, differential flatness, Model Predictive Control (MPC), Mixed-Integer Programming (MIP) are adapted and enhanced. Using these theoretical notions, the thesis concentrates on understanding the geometrical properties of the multi-agent group formation and on providing a novel synthesis framework which exploits the group structure. In particular, the formation design and the collision avoidance conditions are casted as geometrical problems and optimization-based procedures are developed to solve them. Moreover, considerable advances in this direction are obtained by efficiently using MIP techniques (in order to derive an efficient description of the non-convex, non-connected feasible region which results from multi-agent collision and obstacle avoidance constraints) and stability properties (in order to analyze the uniqueness and existence of formation configurations). Lastly, some of the obtained theoretical results are applied on a challenging practical application. A novel combination of MPC and differential flatness (for reference generation) is used for the flight control of Unmanned Aerial Vehicles (UAVs).