Mathematical Programming for Multi-Vehicle Motion Planning Under Communication Constraints

A Thesis

Submitted to the Faculty

of

Drexel University

by

Pramod Abichandani

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy

September 2011

Dedications

To my family, especially my parents, kaka and kaki.

To my advisors, Moshe and Dr. Benson.

To my pain-in-the-neck-but-I-love-her younger sister.

To my friends, especially Gabe.

To my teachers, students, and colleagues.

To the staff members, the janitor, and the front desk personnel.

To the anonymous reviewers of my work.

To each and every one whose path crossed mine.

Above all, to my beloved grandmother whom I miss every single day of my life.

हे दीनबंधु दीनानाथ, मेरी डोर अब तेरे हाथ।

Acknowledgements

Graduate school is a unique place. In my humble opinion, it is an absurd place. The notion of committing half a decade (or more) of one's life to exploring the unknown seems exciting and romantic on the surface. However, it is highly misleading. The realities of grad school are brutal; the research climate is uncertain, the stipend is meager, research papers getting rejected is disheartening, and grading midterm and final exams is mind numbing. So how does one survive graduate school, how does one get a Ph.D.?

Truth be told, there is no straight forward answer to this question, there is no silver bullet. A famous person once said, "You can't connect the dots looking forward. You can only connect them looking backwards". Looking back when I connect the dots, it is apparent that the only reason why I survived grad school is because of the amazing people that I met and the work that I did. I absolutely loved my work and the people that I worked with.

Grad school is a highly personal process, a journey that one needs to take on their own. It is hard to understand this journey without actually going through it. The lessons that I learnt during this incredible journey are innumerable. Rolling with the punches, respecting ideas and opinions without necessarily agreeing with them, and managing uncertainties are just a few of the them. The most important lesson though is the fact that there are no shortcuts in life.

As humbling as these lessons are, ultimately, grad school has given me the confidence that wherever I go, whoever I am with, I will be fine. The dots will all connect. And for this confidence, I sincerely thank you grad school.

Abstract

Mathematical Programming for Multi-Vehicle Motion Planning Under Communication Constraints

Pramod Abichandani

Advisor: Moshe Kam, Ph.D. and Hande Y. Benson, Ph.D.

Multi-Vehicle Motion Planning (MVMP) problems feature multiple vehicles traversing in their work space while avoiding collisions with each other and with other obstacles. Real world MVMP problems require the optimization of suitable performance measures under an array of constraints including kinematics, dynamics, communication connectivity, target tracking, and collision avoidance. The general MVMP problem can thus be formulated as a mathematical program (MP). In this thesis we present a mathematical programming framework that captures the salient features of the general MVMP problem. We use state-of-the-art solution algorithms and associated numerical solvers developed by our group to solve MVMP problems using this framework.

To demonstrate the effectiveness of this framework, we investigate a variant of the general MVMP problem, viz. Multi-Vehicle Path Coordination wherein we generate time optimal velocity profiles for multiple robotic vehicles confined to move along predetermined and fixed paths. Each robot must follow a fixed and known path, arrive at its goal as quickly as possible (or at least not increase the time for the last robot to arrive at its goal) and stay in communication with other robots in the arena throughout its journey. We enforce a variety of communication connectivity constraints that incorporate deterministic and stochastic physical layer communication models to ensure that the robots can communicate with each other while in transit. We develop Partition Elimination constraints that assist in ensuring that the communication network is fully connected while the robots are in transit. The resulting mathematical

programming models are solved using state-of-the-art highly efficient mixed integer nonlinear optimization tools developed by our group. Several conditions that affect the feasibility of the problem are identified and formalized.

Both centralized and decentralized formulations are studied to demonstrate (i) the effect of communication connectivity requirements on robot velocity profiles; (ii) the dependence of the scenario completion time on communication connectivity requirements; (iii) the dependence of computation time on the number of robots; (iv) the tradeoff between the arrival time and the communication connectivity requirements.

As MP solution algorithms and associated numerical solvers continue to develop, we anticipate that MP solution techniques will be applied to an increasing number of MVMP problems and this thesis may serve as a guide for future MVMP research.

Table of Contents

Abstract ii

List of Tables viii

1. Introduction 1

    1.1 Problem Statement 1

    1.2 Contributions 3

    1.3 Organization 4

2. Relevant Background 7

    2.1 Common features of MVMP problems 7

    2.2 Variations of the general multiple robot motion planning problem 10

    2.3 Relevant elements of MVMP problems 11

    2.4 Mathematical Programming (MP) 22

3. Multi-Vehicle Motion Planning using Mathematical Programming 26

    3.1 Representative Works 26

    3.2 General Mathematical Programming Structure of MVMP problems 29

    3.3 Observations and Design Considerations 35

4. Centralized Multi-Vehicle Path Coordination under Communication Connectivity Constraints 39

    4.1 Practical Applications 40

    4.2 Model Elements 40

    4.3 Centralized Formulations 45

    4.4 Feasibility Considerations 58

5. Decentralized Multi-Vehicle Path Coordination under Communication Connectivity Constraints 63

    5.1 Model Elements 64

    5.2 Decentralized Formulation 65

5.3 Algorithm 69
6. Numerical Results 71
6.1 Simulation Setup 71
6.2 Model Implementation Information 72
6.3 Scenarios Tested 76
6.4 Decentralized Scenarios Tested 80
7. Conclusions and Future Directions 86
7.1 Summary of work 86
7.2 Future Directions 88
Appendices 91
A. NLP reformulation 92
A.1 Model Formulation 92
A.2 Simulation Setup 97
Bibliography 121

List of Figures

1.1 General Trajectory Planning Problem 2

1.2 General Trajectory Planning Problem Solution 2

2.1 A non-holonomic four wheeled car-like vehicle 8

2.2 Manifestation of non-holonomic constraints 9

2.3 Kinematics of a two-wheeled differential drive vehicle 12

2.4 Kinematics of a car-like vehicle 13

3.1 Polygonal approximation of a circle 31

3.2 Discretization of space 32

3.3 Target tracking visibility graph 34

4.1 Individual Splines 43

4.2 Spline Curve Construction 43

4.3 An infeasible d_{safe} value 58

4.4 Infeasible Paths 60

4.5 Infeasible Set 62

6.1 Centralized Formulation: A 2 robot case 77

6.2 Decentralized Formulation: A 2 robot case with varying n_{conn} 79

6.3 Decentralized Formulation: Effect of n_{conn} on velocity profiles for a 2 robot case 80

6.4 Decentralized Formulation: A 6 robot case with varying n_{conn} 82

6.5 Decentralized Formulation: A 10 robot case with varying n_{conn} 84

A.1 NLP: A 2 robot scenario with varying communication constraint. Effect of communication constraint on the velocity profiles in a 2 robot scenario.. 100

A.2 NLP : A 6 robot scenario with varying communication constraint 102

A.3 NLP: Effect of communication constraint on the velocity profiles in a 6 robot scenario 103

A.4 NLP: A 10 robot scenario with varying communication constraint 104

A.5 NLP: Effect of communication constraint on the velocity profiles in a 10 robot scenario 105

A.6 A 2 robot scenario with and without the penalty term σ for $n_{conn} = 1$ 107

A.7 NLP: A 4 robot scenario with and without the partition elimination constraints 109

A.8 NLP: Velocity Profile of the robot 2 in absence and presence of the partition elimination constraints and jammer 110

A.9 NLP: A 10 robot scenario with and without the partition elimination constraints 111

A.10 NLP: Velocity profiles of Robot 2 and Robot 3 in a 10 robot scenario with and without the partition elimination constraints in absence and presence of jammers 112

A.11 NLP: A 20 robot scenario with and without the partition elimination constraints 114

A.12 NLP: A 50 robot scenario with and without the partition elimination constraints 116

A.13 NLP: A 10 vehicle scenario in absence and presence of 4 jammers traveling at constant speed. The speeds of the robots change in order to avoid being jammed. 120

List of Tables

2.1 Path primitives used to represent robot paths. 16

3.1 Representative works on MVMP using Mathematical Programming, indication of inclusion of Kinematics $\mathcal{K}$, Dynamics $\mathcal{D}$, Collision Avoidance $\mathcal{C}$, Communication Θ, and other constraints $\mathcal{O}$. 27

6.1 Problem size comparison for centralized and decentralized MINLP formulations for n robots and t_{hor} time steps 72

6.2 Centralized MINLP model parameter values used for simulations 76

6.3 Decentralized MINLP model parameter values used for simulations 80

6.4 Decentralized MINLP Formulation: T_{comp} in seconds for various scenarios . 85

A.1 NLP model parameter values used for simulations 98

A.2 NLP: Effect of n on T_{comp} and Problem Size Reduction 118

1. Introduction

1.1 Problem Statement

1.1.1 Multi-Vehicle Motion Planning

Multi-Vehicle Motion Planning (MVMP) draws on ideas from Mechanics, Computational Geometry, Algorithmic Decision Theory, Control Theory, and Mathematics. In its most general form, motion planning of a mobile robot is defined as follows: *Given an initial position and orientation and a goal position and orientation of a robot in its workspace, generate a trajectory specifying a continuous sequence of positions, orientations, and speeds while avoiding contact with any obstacles. The trajectory starts at the initial position and orientation and terminates at the goal position and orientation. Report a failure if no such trajectory exists* [1]. Figure 1.1 is an example of a general motion planning scenario with multiple robotic vehicles with given initial positions and goal locations. The environment also consists of static obstacles. Figure 1.2 illustrates a set of possible solution trajectories for the robots.

This problem, also called the *kinodynamic motion planning* problem [2], takes into account the kinematics, dynamics, and collision avoidance requirements of the robot and becomes much more difficult to formulate and solve once extended to coordinated motion of multiple mobile robots amidst multiple stationary and moving obstacles [3–5]. The fact that these robotic vehicles share a common workspace raises several challenges. The coordination techniques must take into account the kinematics (non-holonomic for the most part) and dynamics of the robotic vehicles. In addition, they must maintain reliable communication connectivity between robots while ensuring that they do not collide with each other as well as other stationary and moving obstacles. In case of a communication failure and/or collisions, the robots

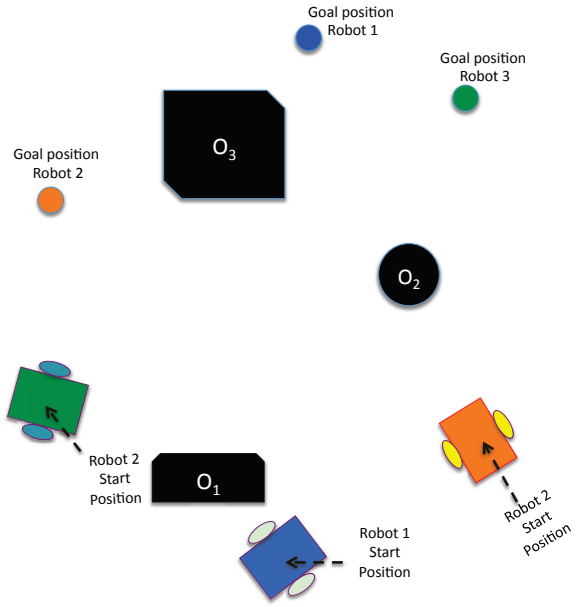

Figure 1.1: Trajectory planning problem: robots in their workspace, black blocks indicate obstacles

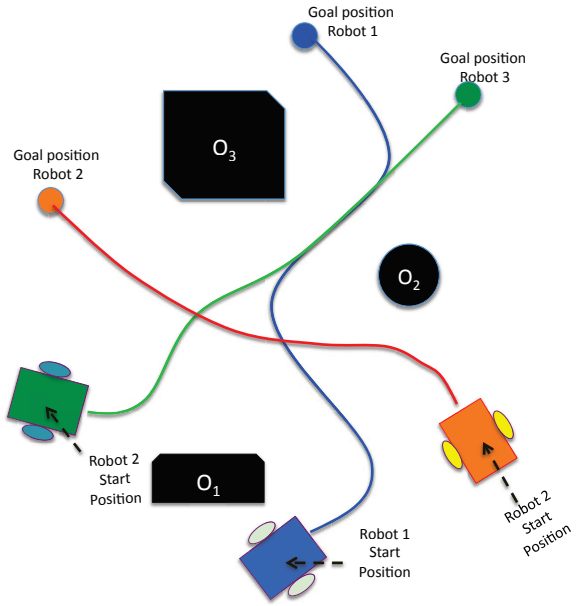

Figure 1.2: Solution trajectories that satisfy kinematics, dynamics, and collision avoidance

must be able to quickly re-plan their motion such that communication is restored and safe motion is made possible. Coordinated kinodynamic motion planning of multiple robots is a relatively new field. The first textbook on this topic [1] was published less than two decades ago and to date, only a handful of textbooks cover it in detail [6, 7]. Coordinated motion planning of multiple robotic vehicles can be used to accomplish several complex tasks with relatively high efficiency. Example applications include automated guidance of ground, underwater, and air vehicles; air traffic control; and manufacturing cell operations.

The work documented in this thesis commenced with the goal of using modeling techniques and numerical solution tools made available through the field of Mathematical Programming (MP) to address the general problem of MVMP under communication connectivity constraints.

Specifically, we study the problem of *fixed-path coordination of multiple robotic vehicles under communication connectivity constraints* which is defined as follows: *Given a group of path-constrained vehicle robots that have fixed and known initial and goal locations, generate time-optimal velocity profiles that satisfy kinematic, dynamic, collision avoidance and communication connectivity constraints.*

Quite often in real world scenarios, robots do not have the liberty of planning arbitrary paths and must follow fixed roads. We present centralized and decentralized formulations of this problem and formulate different communication constraints.

1.2 Contributions

1. We develop several **communication connectivity constraint formulations** using physical layer communication models that capture the deterministic and stochastic nature of inter-robot wireless communication.

2. We develop and solve **centralized and decentralized formulations** for Multi-

Vehicle Path Coordination under communication constraints.

(a) The centralized formulations are Mixed Integer Nonlinear Programming (MINLP) problems. For the case with stochastic communication connectivity constraints, we develop a robust MINLP formulation. This robust MINLP model allows for motion planning of multiple vehicles while enforcing an upper bound on the probability of unsuccessful inter-robot communication communication. We also develop equivalent Nonlinear Programming (NLP) formulations.

(b) For the decentralized framework we develop a Receding Horizon Mixed Integer Nonlinear Programming (RH-MINLP) formulation that is embedded in a sequential decentralized decision making algorithm.

3. For the centralized case, we develop an efficient **Network Partition Elimination** algorithm to ensure that there are no communication network partitions while the robots are in transit. We characterize the computational improvements realized due to the use of this algorithm.

4. We define and formalize several **conditions that can cause infeasibility** in the problems studied, and prove that in absence of the communication connectivity constraints the problem of Multi-Vehicle Path Coordination is always feasible.

5. We **characterize the practical performance** of the numerical solvers MILANO [8] and LOQO [9] in solving MVMP problems by enlisting solution computation times.

1.3 Organization

The remainder of the thesis is organized as follows:

- In **Chapter 2**, we introduce the relevant features, properties, variations, and elements of MVMP problems. Properties and types of mathematical programs along with different solution algorithms and modeling environments used in this thesis are discussed.

- In **Chapter 3**, we present a general MP based framework that captures the salient features of almost all MVMP problems. We examine in detail representative works that demonstrate the application of this framework to the formulation and solution of MVMP problems and summarize several other related works. Finally, we list several design considerations for MVMP using MP.

- In **Chapter 4** and Appendix A, we present centralized formulations for Multi-Vehicle Path Coordination under communication constraints. We test scenarios involving up to fifty (50) robots under both deterministic and stochastic communication constraints. Furthermore, we develop *Partition Elimination* constraints that assist in ensuring that the communication network is fully connected (no network partitions) while the robots are in transit and there exists a communication path from every robot to every other robot (possibly using intermediary robots as relay stations).

- In **Chapter 5**, we develop a decentralized framework for Multi-Vehicle Path Coordination under communication constraints. The framework uses a discrete time Receding Horizon Mixed Integer Nonlinear Programming (RH-MINLP) formulation that is embedded in a sequential decentralized decision making algorithm. We test scenarios involving up to ten (10) robots under both deterministic and stochastic communication constraints.

- Extensive numerical results for different scenarios are presented in **Chapter 6** for both the centralized and decentralized formulations. Information about

the number of decision variables and constraints as a function of the number of robots is provided to indicate the difference in the centralized and decentralized problems. Solution computation times for the scenarios tested are presented.

- Finally, in **Chapter 7**, we summarize our work and present future directions of this research.

2. Relevant Background

Both MVMP and MP are active fields of research. In this chapter, we introduce concepts that are relevant to the development in the following chapters.

2.1 Common features of MVMP problems

While several variations and simplifications of the general motion planning problem have been investigated in the literature, certain features of this problem remain common across all motion planning problems variants.

1. **Holonomic and Non-holonomic motion planning**

   Holonomic system: When all degrees of freedom of a robot can be changed independently (like in a fully actuated robotic arm) the system is called a holonomic system. The problem of motion planning of holonomic robots is called holonomic motion planning problem. A collision-free path is found by building a sequence of collision-free configurations of robots. In a holonomic system, all collision-free configurations of the robot are feasible to achieve [10].

   Non-holonomic system: When the degrees of freedom of a robot system are not independently actuated, the system is called non-holonomic and the corresponding motion planning problem is called non-holonomic motion planning. These constraints usually arise due to the kinematics of the robot system and are characterized by the fact that the system has fewer control variables than configuration variables. For instance, consider a car-like robot shown in Figure 2.4. This robot has two controls (linear and steering velocities) while it moves in a 4-dimensional configuration space. x, y, θ, and ϕ are the configuration variables of the robot that completely specify the robot location with respect

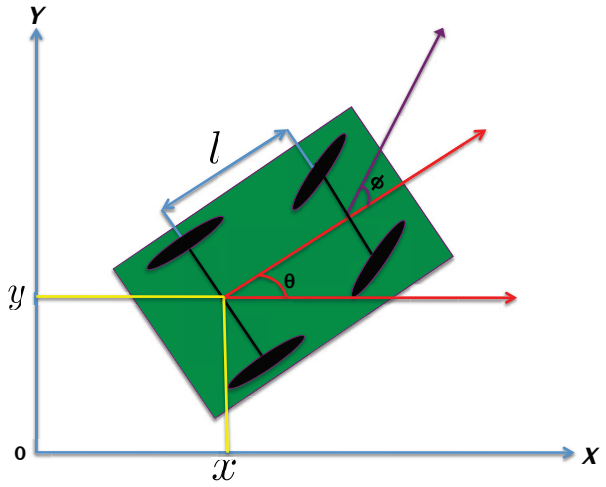

Figure 2.1: Four wheeled rear wheel drive car-like vehicle - x, y, θ, and ϕ are the configuration variables

to the global (X, Y) coordinate system. Due to this limitation, the car cannot rotate around its axis without changing its position. Unlike a holonomic system, for a non-holonomic system, an arbitrary sequence of configurations may not be feasible to achieve. This is a fundamental issue for most type of mobile robotic systems [10, 11]. As shown in Figure 2.2, because of the non-holonomic constraint that disallows the car-like vehicle to slide sideways, the vehicle has to perform a parallel parking maneuver.

2. **Cooperative and Non-cooperative motion planning**

As the name suggests, cooperative motion planning refers to a scenario wherein the robots cooperate with each other by sharing information about their motion plans and intentions, where as in non-cooperative schemes, robots do not share any information and may even compete with each other. Cooperative planning algorithms have been developed for systems involving Unmanned Aerial Vehicles (UAVs) [12, 13], Autonomous Underwater Vehicles (AUVs) [14], and ground robots [15]. Non-cooperative approaches include use of concepts from game

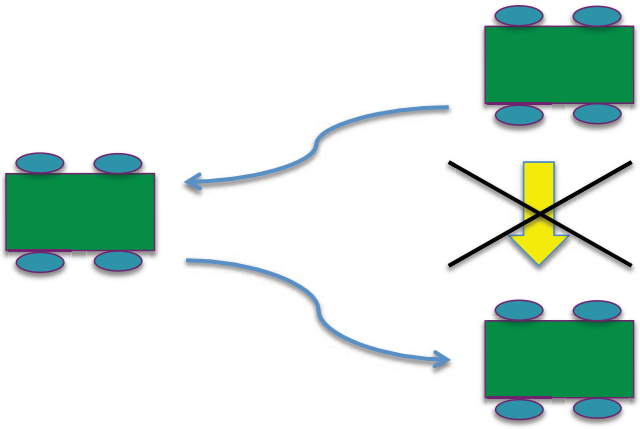

Figure 2.2: A parallel parking maneuver is necessitated because the non-holonomic constraints of the car-like vehicle disallow it from sliding sideways

theory for multiple robot motion planning [16].

3. **Centralized and Decentralized motion planning**

   In a centralized motion planning scenario, all motion plans are made by a central coordinator and decision maker [17–19]. Decentralized planning involves each robot making its own motion plan decisions [13,20]. Centralized planning solves the motion planning problem for all robots together. While centralized planners allow for possible global optimization and completeness, the dimension of the feasible region grows rapidly with the number of robots. This growth can lead to increased computational burden, making centralized planning more suitable for offline computations. In a decentralized framework, each robot only solves its own problem and so the dimension of the feasible region grows linearly in the number of robots. This property makes decentralized formulations more suitable for online real-time planning. However, decentralized planning often does not offer guarantees of global optimality. In [21], the authors present a motion planning framework that attempts to exploits the benefits of both

centralized and decentralized planning simultaneously.

2.2 Variations of the general multiple robot motion planning problem

By adding constraints to the general problem, several simplified versions are obtained.

1. **Path Planning** deals with the problem of finding a collision-free geometric path that is represented using different primitives such as straight lines, circular arcs, and splines. Approaches such as visibility graphs [22], exact cell decomposition [23, 24] and approximate cell decomposition [25–29], Voronoi diagrams [30, 31], potential fields [32–34], navigation functions [35], and randomized path planning [36–39] have been used to address this problem.

2. **Trajectory Planning** deals with the problem of finding a spatio-temporal solution, i.e. a collision-free geometric path and position along the path at specific time instances (and hence velocities along the path). Several approaches including those used for path planning have been developed/extended to address trajectory planning issues. These include potential fields [40–42], genetic algorithms [43], randomized planners [44, 45], mathematical programming based planners [13, 46–48], iterative bargaining schemes [49], game-theoretic frameworks [50], dynamic sensing and communication networks [21], and search algorithms [51] among others.

3. **Path Coordination** deals with multiple robots with fixed paths coordinating with each other so as to avoid collisions and reach their respective destination points. While several approaches have been used to address the problem of path coordination of multiple robots, this is a relatively untouched area. Approaches reported upon in the literature include use of coordination diagrams [52], con-

strained configuration space roadmap [53], and grouping robots with shared collision zones into subgroups [54]. In [19], mixed integer linear programming (MILP) formulations were used to generate continuous velocity profiles for a group of robots that satisfy kinodynamics constraints, avoid collisions and minimize the task completion time. In [17, 18, 20], we extended this body of work by adding communication constraints to the problem and using state of the art interior-point methods to solve the resulting nonlinear programming problem.

4. Other variants such as **Target Tracking** [55, 56] and **Formation Control** [57–60] are realized by placing additional constraints on one or more of the above mentioned variants.

2.3 Relevant elements of MVMP problems

To model a MVMP problem effectively, the constraints on motion arising due to the robot body construction (kinematics) and the forces and torques generated by the wheels (dynamics) must be considered. Equally important is the choice of path primitives - the functions used to describe the paths that the vehicles traverse, and the inter-robot wireless communication models.

2.3.1 Robot Kinematics

Mobile robot kinematics are typically non-holonomic in nature. They are expressed in terms of generalized coordinates and generalized velocities. Generalized coordinates are used to uniquely describe any possible configuration of a system relative to a reference configuration [61]. Non-holonomic constraints are Pfaffian constraints that feature differential equations in terms of generalized velocities that are not integrable; these constraints cannot be written as algebraic constraints in terms of the generalized coordinates [10] and are usually written as

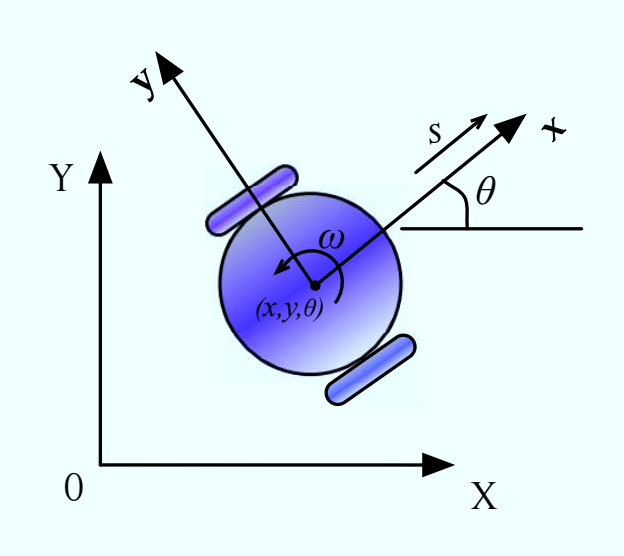

Figure 2.3: Two-wheeled differential drive robot vehicle - x, y and θ are the generalized coordinates

$$C(q)\dot{q} = 0, \tag{2.1}$$

where q is the vector of generalized coordinates, and $C(q)$ is the Pfaffian constraint matrix. The feasible velocities $\dot{q}$ belong to the null space of $C(q)$.

For a two-wheeled differential drive mobile robot as shown in Figure 2.3, the kinematic model is represented by (2.2) - (2.4). The associated non-holonomic constraint that prevents the robot from sliding sideways is expressed as (2.5).

$$\dot{x} = s\cos(\theta) \tag{2.2}$$

$$\dot{y} = s\sin(\theta) \tag{2.3}$$

$$\dot{\theta} = \omega \tag{2.4}$$

$$\dot{x}sin(\theta) - \dot{y}cos(\theta) = 0 \tag{2.5}$$

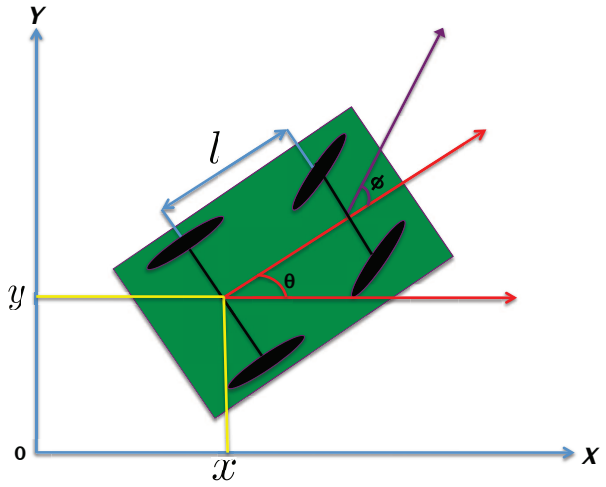

Figure 2.4: Rear-wheel drive car-like vehicle - x, y, θ, and ϕ are the generalized coordinates

Here s and ω are the linear and angular speeds of the robot respectively. x, y and θ are the generalized coordinates of the robot with respect to the global (X, Y) coordinate system.

For a rear-wheel drive car-like robot shown in Figure 2.4, the kinematic model is given by (2.6) - (2.9),

$$\dot{x} = s\cos(\theta) \tag{2.6}$$

$$\dot{y} = s\sin(\theta) \tag{2.7}$$

$$\dot{\theta} = s\frac{\tan\phi}{l} \tag{2.8}$$

$$\dot{\phi} = \Xi \tag{2.9}$$

where s and Ξ are the linear and steering speeds of the vehicle respectively. x, y, θ, and ϕ are the generalized coordinates of the robot with respect to the global (X, Y) coordinate system. The non-holonomic constraint associated with the front pair

of wheels is given by (2.10). The non-holonomic constraint associated with the rear pair of wheels is given by (2.11).

$$\dot{x}\sin(\theta+\phi) - \dot{y}\cos(\theta+\phi) - \dot{\theta}l\cos\phi = 0 \tag{2.10}$$

$$\dot{x}sin(\theta) - \dot{y}cos(\theta) = 0 \tag{2.11}$$

2.3.2 Robot Dynamics

A considerable amount of motion planning literature builds on kinematic models of mobile robots. When the speeds and accelerations of the robots are slow, and the loads experienced by the robots are relatively low, the assumptions of "perfect velocity tracking" and "no slipping" are reasonable. In such cases, explicit consideration of vehicle dynamics can be avoided. At high speeds and load values, robot dynamics cannot be avoided and need to be modeled.

Dynamic models of mobile robots are derived using Lagrange's equation of motion by considering the mass and inertia of the driving wheels. Lagrange's equation of motion (2.13) features Lagrangian function $\mathcal{L}$ which is defined by (2.12) as the difference between the Kinetic Energy $\mathcal{V}$ and Potential Energy $\mathcal{U}$ of a system [62].

$$\mathcal{L}(q,\dot{q}) = \mathcal{V} - \mathcal{U}(q) \tag{2.12}$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\partial q_i} = \mathcal{F}_i + \Psi_i \tag{2.13}$$

Here q is the vector of generalized coordinates q_i, $i = 1 \ldots \xi$, and $\mathcal{F}$ is the sum of all external input forces, disturbances, and dissipative effects acting on the body, and Ψ_i is the component of constraint force acting along the i^{th} coordinate that arise

due to the holonomic and non-holonomic constraints of the mobile robot.

The equations of motion for a system with holonomic and non-holonomic constraints can be described by

$$M(q)\ddot{q} + F(q,\dot{q})\dot{q} + K(\dot{q}) + G(q) + \chi = B(q)d + C(q)^T\lambda. \tag{2.14}$$

Here $M(q)$ is a symmetric, positive definite inertia matrix, $F(q,\dot{q})$ is the matrix of centripetal and Coriolis acceleration terms, $K(\dot{q})$ is the surface friction, $G(q)$ is the gravitational force, and χ represents the unknown disturbances including unstructured unmodeled dynamics. d is the input vector and $B(q)$ is the input transformation matrix. λ is the vector of Lagrange multipliers and $C(q)^T\lambda$ represents generalized forces related to the holonomic and non-holonomic kinematic constraints (2.1) [63–67].

2.3.3 Path Primitives

Path primitives are parametric functions that represent paths of robots in their workspace. In $\Re^2$, they are images of curves $\mathcal{P}$ that are defined by the map

$$\mathcal{P} : [\varpi_0, \varpi_1] \rightarrow \Re^2, \quad \varpi \rightarrow \mathcal{P}(\varpi) = [\alpha(\varpi)\beta(\varpi)]^T, \tag{2.15}$$

under the vectorial function $\mathcal{P}(\varpi)$.

The arc length u along the path can be evaluated by the relationship

$$u = \int_{\varpi_0}^{\varpi_1} ||\dot{\mathbf{p}}(z)||dz. \tag{2.16}$$

Here $||\cdot||$ denotes the Euclidean norm.

The seminal results on the shortest paths for car-like robots by Dubins [68] and Reeds and Shepp [69] use straight lines and circular arcs as path primitives. The

optimality criterion in these works was minimal curvature along the path. While the resulting paths were optimal in this sense, they suffered from discontinuous curvature at the points where the circular arcs joined with the straight line-segments. For two-wheeled differential drive mobile robotic vehicles with non-holonomic constraints, the path primitives should have continuous first (slope) and second (curvature) derivatives along the entire path for allowing smooth motion; i.e., they should be G^2 paths [70]. Furthermore to avoid slipping, the curvature should be continuously differentiable (G^3 path) [70]. For car-like robots, the derivative of the curvature should also be bounded[1] and continuous since the steering velocity is bounded. If there is a discontinuity along the path, the vehicle needs to stop to reorient its wheels.

To overcome the jerks that would occur due to these discontinuities, smoothing primitives such as clothoids, cubic spirals or polynomial curves have been used. Primitives such as Quintic polynomials, B-splines, and polar splines have a closed form expression for their coordinates, where as primitives such as clothoids, cubic spirals, intrinsic spline are parametric curves whose curvature is a function of their arc length [71]. Table 2.1 summarizes different primitives that have been used in the literature.

Table 2.1: Path primitives used to represent robot paths

Name	**Path Primitives**	**Features**
Dubin [68]	Straight lines and circular arcs	Minimum curvature paths, discontinuity in curvature
Continued on next page		

[1]The curvature should be less than 1 for Reed and Shepp and Dubins cars [10]

Table 2.1 – continued from previous page

Name	**Path Primitives**	**Features**
Reeds and Shepp [69]	Straight lines and circular arcs	Minimum curvature paths with cusps allowed
Froissart and Mechler [72]	Cartesian polynomials	Polynomial functions of parameter, polynomial of at least fifth order should be used for continuous curvature. Maximum value of curvature is difficult to compute
Gallina and Gasperetto [73]	Parametric sum of harmonics	Sums of sine and cosine functions, continuous derivates of whatever order, closed form expressions available
Fleury et al. [74]	Clothoids and Anticlothoids	Proven to be time-optimal trajectories for two-wheeled mobile robots. No closed-form expression of the position along the curve is available [73]
Berglund et al. [75]	B-Splines	Smooth closed form splines, quartic B-splines can be designed to have continuous derivatives of curvatures

Continued on next page

Table 2.1 – continued from previous page

Name	**Path Primitives**	**Features**
Nelson [76, 77]; *Takahashi* [78]; *Pinchard et al.* [79]	Quintic Curves and Polar Splines	Computationally simple, closed-form expressions, provide continuous curvature and precise matching of the boundary conditions at the line-curve junctions on the paths; minimize jerk (derivative of acceleration)
Kanayama and Hartman [80]	Cubic Spiral	Robot path specified using posture (position and orientation) instead of sequence of curve segments Smoother curvature than clothoids. No closed form expressions for positions along the curve available
Delingette et al. [81]	Intrinsic Spline	Curves with polynomial curvature profile
Fraichard and Scheuer [71]	CC Steer method	Paths comprised of straight lines, circular arcs, and clothoids; have continuous bounded curvature and bounded curvature derviative
Piazzi *et al.* [70]	η^2 and η^3 splines	Unified framework to generate or approximate, a variety of curve primitives such as circular arcs, clothoids, spirals, etc.

2.3.4 Inter-robot communication

We consider robots that are equipped with a wireless transceiver. The body of literature that addresses wireless communications in networks formed of mobile vehicles is very broad and continuously evolving. We focus our attention on physical layer communication models that explicitly model the received signal power as a function of the transmitted signal power, channel conditions, Euclidean distance between the transmitter and receiver pairs, and the random attenuations caused due to environmental factors.

Consider two robots that try to communicate with each other at a given point in time. The Euclidean distance between them is denoted by d. The signal transmission power of the wireless transceiver placed on the transmitter robot is denoted by P_t. The received signal power of the wireless transceiver placed on the receiver robot is denoted by P_r. The power experienced by the receiver robot node is calculated using Friis's equation [82]

$$P_r = P_t G_t G_r \left(\frac{\lambda}{4\pi d}\right)^{\alpha} \tag{2.17}$$

where α is the path loss exponent. The values of α range from 1.6 (indoor with line of sight) to 6 (outdoor obstructed) depending on the environment. λ is the wavelength and is equal to c/f, where $c = 3$ x 10^8 m/s. Here we will use $f = 2.4$ x 10^9 Hz. The values of G_t and G_r (antenna gains) are assumed to be 1.

Using Friis's equation, the average large-scale path loss between a transmitter receiver pair can be expressed a function of distance as

$$\overline{PL}(d) \propto \left(\frac{d}{d_0}\right)^{\alpha} \tag{2.18}$$

When expressed in decibels (dB), we can rewrite (2.18) as

$$\overline{PL}(d) = \overline{PL}(d_0) + 10\alpha \log\left(\frac{d}{d_0}\right) \tag{2.19}$$

In (2.18) and (2.19), d_0 is the close-in reference distance that is in the far field of the antenna so that near-field effects do not alter the reference path loss. d_0 is calculated using the Friis's equation such that $d \geq d_0 \geq d_f$, where d_f is the far-field distance. d_f is calculated as

$$d_f = \frac{2D_{dim}^2}{\lambda} \tag{2.20}$$

where D_{dim} is the largest physical linear dimension of the antenna. $d_f \gg D_{dim}$ and $d_f \gg \lambda$ for far field regions. The bars in (2.18) and (2.19) denote the ensemble average of all possible path loss values for a given value of d.

For a wireless node with a 3 cm long antenna that uses a 2.4 GHz frequency, d_f can be calculated using (2.20) as indicated by (2.21)

$$d_f = \left(\frac{2(0.03)^2}{\left(\frac{3.8 \times 10^8}{2.4 \times 10^9}\right)}\right) \approx 0.011 \text{ meters} \tag{2.21}$$

Other than the path-loss, shadowing effects caused due to presence of obstacles in the environment cause random attenuation in wireless communication. Shadowing effects are represented by a log-normal random variable that is factored into (2.18) as indicated by

$$\overline{PL}(d) \propto \left(\frac{d}{d_0}\right)^{\alpha} \Upsilon, \tag{2.22}$$

where Υ has a pdf

$$f_{\Upsilon}(x) = \frac{10/\ln 10}{\sqrt{2\pi}\upsilon x} exp\left\{-\frac{(10\log_{10}x)^2}{2\upsilon^2}\right\}, x \geq 0. \tag{2.23}$$

When expressed in decibels (dB), the path-loss can be calculated as

$$PL(d)[dB] = \overline{PL}(d_0) + \Upsilon_{\upsilon} = \overline{PL}(d_0) + 10\alpha\log\left(\frac{d}{d_0}\right) + \Upsilon_{\upsilon} \tag{2.24}$$

where Υ_{υ} is the zero-mean Gaussian random variable with standard deviation υ. The received power $P_r(d)$ is expressed as

$$P_r(d)[dBm] = P_t[dBm] - PL(d)[dB] \tag{2.25}$$

The Signal to Noise Ratio (SNR) experienced by the receiver robot is calculated using the relationship

$$SNR = P_r/N \tag{2.26}$$

where N is the noise experienced by receiver robot. The noise is assumed to be thermal $N = kTBF$. k is the *Boltzmann's constant* given by 1.38×10^{-23} Joules/Kelvin and B is the equivalent bandwidth of the receiver. T is the ambient room temperature (typically 290 Kelvin to 300 Kelvin) and F is noise figure of the receiver.

The *outage probability* $\mathbb{P}_0$ is defined as the probability that the SNR experienced is less than a certain threshold η_{snr}, or equivalently the received power P_r is below a certain threshold η_c.

$$\mathbb{P}_0 = \mathbb{P}(\text{SNR} < \eta_{snr}) = \mathbb{P}(P_r < \eta_c) \tag{2.27}$$

Interested readers are referred to [83] and its references for further information about physical layer communication models.

2.4 Mathematical Programming (MP)

In its most general form, a mathematical program (2.28) minimizes (or maximizes) an objective function $\Phi(\zeta, \sigma)$ subject to a set of constraints $\Omega(\zeta, \sigma) \leq 0$, where $\zeta \in \mathbb{R}^{n_1}$ and $\sigma \in \mathbb{Z}^{n_2}$ are continuous and discrete decision variables, respectively, and $\Omega : \mathbb{R}^{n_1} \times \mathbb{Z}^{n_2} \rightarrow \mathbb{R}^m$

$$\begin{aligned} \text{minimize} \quad & \Phi(\zeta, \sigma) \\ \text{subject to} \quad & \Omega_q(\zeta, \sigma) \leq 0, \quad q = 1 \ldots m \\ & \zeta \in \mathbb{R}^{n_1}, \sigma \in \mathbb{Z}^{n_2} \end{aligned} \tag{2.28}$$

Mathematical programming frameworks offer the flexibility to accommodate multiple complex constraints simultaneously. Once the constraints have been formulated, an appropriate performance measure can be constructed to be used as the objective function.

2.4.1 Relevant MP classes and constraint types

The constraints and objective function in (2.28) can be linear or nonlinear functions of the decision variables. If they are non-linear functions, they can be convex or non-convex functions. Additionally, the decision variables can be discrete and/or continuous. Accordingly, different types of MP classes can be constructed. When the constraints as well as Φ are linear functions of ζ, (2.28) results in a Mixed Integer Linear Programming (MILP) problem. When $n_2 = 0$ and at least one constraint or Φ is a non-linear function of ζ, (2.28) results in a Nonlinear Programming (NLP) problem. MINLPs are nonlinear programming problems that feature both continuous and discrete variables. The models that we have developed can be broadly classified

as Mixed Integer Nonlinear programs (MINLP) with locations, speeds, and accelerations of the robots being continuous variables. We use binary variables to specify the state of communication connectivity between robot pairs. The binary variables are one of the sources of non-convexity in our model. In order to capture the *either-or* (disjunctive) nature of constraints involving the binary variables, we make use of Big-M constraints [84]. An example of a Big-M constraint is discussed.

Big-M constraints

Suppose we have a discrete variable $\mathcal{V}_d$, $\mathcal{V}_d \in \{0, 1\}$, and a non-negative continuous variable $\mathcal{V}_c$, $\mathcal{V}_c \geq 0$, such that, if $\mathcal{V}_d = 0$, then $\mathcal{V}_c = 0$. This disjunctive constraint can be recast as a Big-M constraint by introducing a large valued number $\mathcal{V}_M$ as follows:

$$\mathcal{V}_c - \mathcal{V}_M * \mathcal{V}_d \leq 0 \tag{2.29}$$

where $\mathcal{V}_M$ is a large number.

It is clear that when $\mathcal{V}_d = 0$, (2.29) gives $\mathcal{V}_c \leq 0$, which along with the constraint that $\mathcal{V}_c \geq 0$ results in $\mathcal{V}_c = 0$. When $\mathcal{V}_d = 0$, (2.29) gives $\mathcal{V}_c \leq \mathcal{V}_M$ which will be trivially satisfied for a sufficiently large $\mathcal{V}_M$ value.

It is a well known fact that the main challenge in using Big-M formulations is the proper choice of M value. Selecting sufficiently large value for M is necessary for the constraints to be trivially satisfied. However, larger values can lead to numerical instability in the solution process [85].

2.4.2 Solution Algorithms

The difficulty of solving (2.28) depends on the convexity of the feasible region and objective function, the integrality of the variables, and the size of the problem. Many practical solution algorithms, numerical solvers, and modeling environments have

been developed to handle different types of mathematical programs efficiently [86,87]. Commercial and/or open-source solvers such as `CPLEX` [88], `SeDuMi` [89], `MOSEK` [90], `LOQO` [9], `IPOPT` [91], `MINLP` [92], `SNOPT` [93], `MATLAB`'s `fmincon` and `quadprog`, and others have been written to solve several MP classes.

When $n_2 = 0$, Φ is a convex function of the decision variable ζ, and $\{\zeta : \Omega(\zeta) \leq 0\}$ is a nonempty, closed, and bounded convex set, the resulting MP can be solved in polynomial time with guaranteed globally optimal solutions under mild regularity conditions on the problem [86]. Finding globally optimal solutions for non-convex problems can be challenging. Non-convexities are introduced in the problem (2.28) due to one or more of the following reasons:

1. When $n_2 > 0$ i.e. there are integer variables in the problem.

2. The objective function is a non-convex function of the decision variables.

3. The feasible region formed by the constraints is a non-convex set.

Several large-scale NLP solution techniques can handle non-convexities but do not provide guarantees of finding globally optimal solution. These include solvers like `LOQO` that uses interior-point methods [9], `KNITRO` that uses trust-region algorithms [94], and `SNOPT` that uses a quasi-Newton algorithm [93] among others. These solvers also incorporate mechanisms to detect infeasibilities and unboundedness in the problem. A variety of techniques such as branch-and-bound [95,96], evolutionary algorithms [97], and Adaptive Simulated Annealing (ASA) [98] among others exist for expanding the search for globally optimal solutions.

MINLP solution algorithms have seen considerable progress in the past decade. Typically, the solution approaches use integer programming techniques such as branch-and-bound, generalized Bender's decomposition [99], and outer approximation [100] to handle the integer variables, and active set methods [101] or interior-point methods

to handle the continuous relaxations.

2.4.3 Modeling Environments

Modeling environments such as `AMPL` [102], `GAMS` [103], `MATLAB` and `YALMIP` [104] allow for quick development of models for testing/simulation purposes. In this thesis, we used `AMPL` and `MATLAB`. In particular, `MATLAB` is used extensively since it allows for object oriented programming and functions that are used to generate path information, implement decentralized decision making algorithms, visualize data points, and post process solutions.

3. Multi-Vehicle Motion Planning using Mathematical Programming

In this chapter, we propose a MP based framework that captures the salient features of almost all MVMP problems. We review several publications that use MP concepts and solution tools to model and solve MVMP variants. The body of work that uses MP techniques to address variants of the general MVMP problem continues to grow [12–14, 17–20, 46–49, 56, 105–117]. Table 3.1 provides a short introduction to fifteen (15) pertinent publications. We compare them in terms of the types of constraints they deal with (kinematics, dynamics, collision avoidance, communication, and others). We also note the resulting formulation and the solvers used. Of these fifteen studies, we have selected three (3) for a deeper look. These are studies by *Schouwenaars et al.* [13], *Peng and Akella* [19], and *Derenick et al.* [56]. Collectively, these three works along with the formulations described in Chapters 4 and 5 cover almost all of the features of MVMP problems, viz. kinematics, dynamics, collision avoidance, communication, and visibility/tracking.

3.1 Representative Works

The problems studied in [13], [19], and [56], (as most MVMP problems) can be expressed in the following mathematical program formulation:

Table 3.1: Representative works on MVMP using Mathematical Programming, indication of inclusion of Kinematics $\mathcal{K}$, Dynamics $\mathcal{D}$, Collision Avoidance $\mathcal{C}$, Communication Θ, and other constraints $\mathcal{O}$.

Name	MVMP variant	$\mathcal{K}$	$\mathcal{D}$	$\mathcal{C}$	Θ	$\mathcal{O}$	Formulation and Solver
Schouwenaars et al. [13] and [107]	Discrete time Decentralized Trajectory planning of multiple UAVs with safety guarantee	✓	✓	✓	-	Hard safety at all times	MILP, `CPLEX`
Peng and Akella [19] and [108]	Discrete space, Continuous time Centralized Kinodynamic Fixed Path Coordination of multiple robots	✓	✓	✓	-	-	MINLP, MILP approximations, `CPLEX`
Derenick et al. [56]	Discrete time Centralized Target Tracking of ground robots with communication constraints	✓	✓	✓	✓	Target visibility/tracking at all times	SDP, SOCP, `SeDuMi`, `MOSEK`, YALMIP
Abichandani et al. [17], [18], [20]	Discrete time Centralized and Decentralized Muti-Vehicle path coordination under communication constraints	✓	✓	✓	✓	Connected communication graph at all times	NLP, `LOQO`, AMPL, MATLAB
Inalhan et al. [49]	Discrete time Trajectory planning of multiple UAVs via optimal bargaining process	✓	✓	✓	✓	-	NLP, `fmincon`, MATLAB
Richards and How [109]	Discrete time Centralized Trajectory planning of multiple UAVs	✓	✓	✓	-	-	MILP, `CPLEX`, AMPL, MATLAB
Keviczky et al. [110] and [111]	Discrete time Decentralized Receding Horizon Control and Coordination of Autonomous Vehicle Formations	✓	✓	✓	✓	-	MILP, `CPLEX`, AMPL, MATLAB
Pallottino et al. [112]	Discrete time Conflict Resolution for Multiple Aircrafts in a Shared Airspace	-	✓	✓	-	-	MILP, `CPLEX`
Borrelli et al. [113]	Discrete time Centralized Trajectory Generation of Multiple UAVs	✓	✓	✓	-	-	MILP, NLP `CPLEX`, `IPOPT`
Singh and Fuller [114]	Discrete time Centralized Trajectory Generation of a single UAV - extendable to multiple UAVs	✓	✓	-	-	-	QP, `quadprog`, MATLAB
Aoude et al. [105]	Discrete time Multiple Spacecraft Reconfiguration Maneuvers	✓	✓	✓	-	Pointing restrictions	NLP, `SNOPT`

$$
\begin{aligned}
\textbf{minimize} \quad & \textbf{Objective Function } \Phi(\zeta, \sigma) \\
\textbf{subject to} \quad & \\
& \textbf{Kinematics } \mathcal{K}(\zeta, \sigma) \leq 0 \\
& \textbf{Dynamics } \mathcal{D}(\zeta, \sigma) \leq 0 \\
& \textbf{Collision-avoidance } \mathcal{C}(\zeta, \sigma) \leq 0 \\
& \textbf{Communication } \Theta(\zeta, \sigma) \leq 0 \\
& \textbf{Other Constraints } \mathcal{O}(\zeta, \sigma) \leq 0
\end{aligned}
\tag{3.1}
$$

Schouwenaars et al. in [13] present a cooperative decentralized algorithm for trajectory generation of multiple UAVs with hard safety guarantees. Of main interest in this work is the MILP formulation of the trajectory planning problem, and a decentralized receding horizon decision making algorithm. This algorithm takes into account the trajectories/plans of other UAVs and maintains a guaranteed safe plan by ensuring that the trajectories of all UAVs terminate in non-intersecting circular paths (called loiter circles). The work presented in this paper led to one of the first practical implementations of MP-based motion planning of UAVs [107].

In [19], *Peng and Akella* deal with the problem of collision-free coordination of multiple robots with constraints on kinematics and dynamics along specified paths such that the traversal time of the set of robots is minimized. The solution of this problem is a time schedule for each robot along its path. Each robot's path is divided into segments. Each segment is then checked for a possibility of collisions and is accordingly labelled as collision segment or collision-free segment. The authors use a well established result obtained from optimal control theory (minimum and maximum time control of a double integrator with inequality constraints on control (accelera-

tion) and state (velocity and hence position) [118]) to obtain closed form formulae for minimum and maximum times that the double integrator should take to traverse a path segment of given length. Using these closed form expressions as constraints on robot path segment traversal times, the authors construct a MINLP with additional constraints on robot vehicle kinematics, dynamics, and collision avoidance.

In [56], *Derenick et al.* formulate a target tracking by multiple robots problem as a discrete-time generic semidefinite program (SDP) to yield an optimal robot configuration over a given time step. The framework guarantees that the target is tracked by at least a single robot while the robots maintain full communication connectivity with each other. The authors use a key property of the Laplacian matrix of a weighted graph, namely that the second smallest eigenvalue of the Laplacian is a measure of connectivity of the graph. This property is used to formulate Linear Matrix Inequalities (LMIs) [119] involving Laplacian matrices of graphs that represent target visibility and inter-robot communication connectivity. This work demonstrates the use of spectral graph theory and semidefinte programming techniques to solve the target tracking problem.

3.2 General Mathematical Programming Structure of MVMP problems

As we stated, in its most general form, a MVMP problem can be expressed by the MP (3.1) and [13], [19], and [56] are excellent examples of the use of (3.1). In the following, we highlight details about the decision variables, objective functions, constraint formulations, and results discussed in [13], [19], and [56].

3.2.1 Decision Variables ζ, σ

Depending on the task to be performed and modeling approach adopted, the decision variables for (3.1) can be reference speed of each vehicle [13], segment traversal

times and speeds at the start of each segment of a discretized path space [19], or positions of the robot vehicles in continuous Euclidean space [56].

3.2.2 Objective Function $\Phi(\zeta, \sigma)$

In the decentralized formulation of *Schowenaars et al.* [13], $\zeta = \mathbf{u}_t$, where $\mathbf{u}_t$ is the reference speed of each vehicle at current time step t, and the objective function for each vehicle $\Phi_a(\zeta)$ is a piecewise linear cost function that comprises of the weighted 1-norm of the difference of the current state $\mathbf{x}_t$ with the desired state $\mathbf{x_f}$ minus the scalar product of the current velocity vector $\mathbf{s}_t$ with the vector indicating the direction from initial position $\mathbf{p_0}$ to the final position $\mathbf{p_f}$. The values of $\mathbf{x}_t$ and $\mathbf{s}_t$ depend on the decision variable $\zeta = \mathbf{u}_t$. q and r are weights that can be tuned appropriately. $\Phi_a(\zeta)$ provides a minimum time formulation.

$$\Phi_a(\zeta) = \sum_{t=1}^{T} [(q'|\mathbf{x}_t - x_f|) - r(\mathbf{p}_t - p_0)'\mathbf{s}_t], \tag{3.2}$$

In [19], the objective function to minimized is the scenario completion time, which is defined as the time taken by the last arriving robot. To maximize the tracking visibility, *Derenick et al.* in [56] minimize the negative of the second smallest eigenvalue of the visibility graph.

3.2.3 Kinematic Constraints $\mathcal{K}(\zeta, \sigma) \leq 0$

To represent vehicle kinematics, *Schouwenaars et al.* in [13] use a discretized velocity control model. *Peng and Akella* use the following double integrator model in [19].

$$\begin{pmatrix} \dot{x} \\ \dot{s} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ s \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} a \tag{3.3}$$

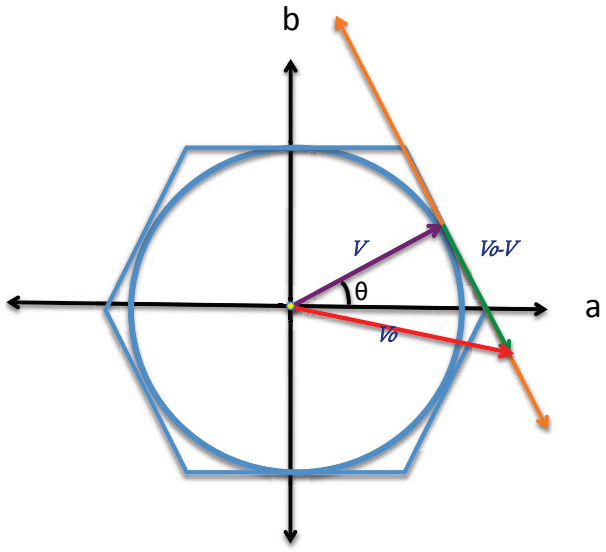

Figure 3.1: Approximation of a circle with radius V_{max} by a polygon - hexagon in this case. The equation of the orange colored line is obtained by the dot product $(\mathbf{V_0} - \mathbf{V}) \cdot \mathbf{V} = 0$

where $x(t)$, $s(t) = \mathrm{d}x(t)/\mathrm{dt}$, and $a(t) = \mathrm{d}s(t)/\mathrm{dt}$ represent the position, speed, and acceleration of the robot, respectively. *Derenick et al.* use a single order fully actuated model in [56].

3.2.4 Dynamic Constraints $\mathcal{D}(\zeta, \sigma) \leq 0$

Schouwenaars et al. in [13], linearize the non-linear dynamic bounds by using N-sided polygonal approximations of a circle that represents dynamic bounds V_{max} on speeds $\mathbf{V_0} = [\dot{x}; \dot{y}]^T$. This introduces N linear constraints for each non-linear constraint. Any point being approximated by the side of the N-sided polygon shown in Figure 3.1 can be represented by

$$\dot{x}cos\theta + \dot{y}sin\theta \leq V_{max} \quad \theta = \frac{2\pi n}{N}, \quad n = 1 \ldots N \tag{3.4}$$

Non-linear bounds on speed and acceleration are used in [19] and [56] (second order

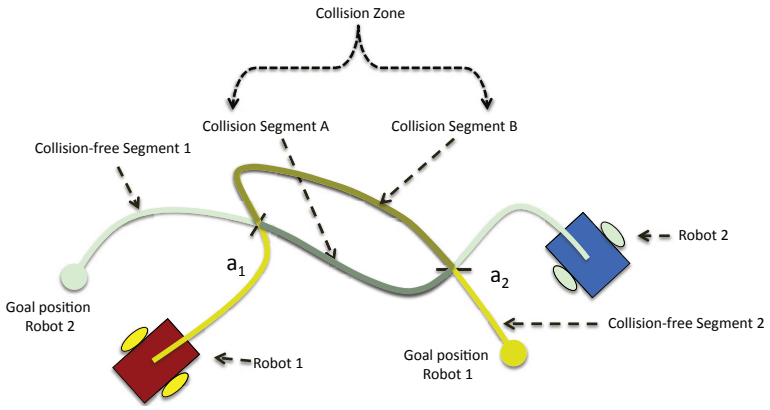

Figure 3.2: Each robot's path is divided into one or more collision segments and collision-free segments. Any pair of points (one each from individual collision segments A and B) between and including points a_1 and a_2 form a collision pair. The individual collision segments A and B together form a collision zone.

conic inequalities). Both [13] and [19] are good examples of using Big-M constraints to express several *either-or* constraints.

3.2.5 Collision Avoidance Constraints $\mathcal{C}(\zeta, \sigma) \leq 0$

Collision avoidance constraints require the robots to avoid colliding with obstacles as well as other robots in the workspace at all times. In [13], *Schouwenaars et al.* approximate the 2-norm distances using 1-norm to reduce complexity, and place an avoidance box of size $2d_{safe}$ around each vehicle, where d_{safe} is the collision avoidance distance. The collision avoidance requirements are then formulated using Big-M constraints.

Discretizing different aspects of the problem changes the nature of these constraints. In [19], by discretizing space instead of time, *Peng and Akella* express collision avoidance constraints by using segment traversal times (Figure. 3.2). The authors note that if two robots i and j could potentially collide if simultaneously traversing segments z_i and z_j respectively, then one of the following must hold to

avoid collisions:

$$\varsigma^j_{z_j} \geq \varsigma^i_{z_i+1} \tag{3.5}$$

$$\varsigma^i_{z_i} \geq \varsigma^j_{z_j+1}, \tag{3.6}$$

where, ς represents segment entry time. The first inequality means that robot i exits segment z_i before robot j enters segment z_j. The second inequality means that robot j exits segment z_j before robot i enters segment z_i

By introducing an arbitrarily large number M, these disjunctive (either-or) constraints can be converted into (3.7).

$$\begin{aligned} \varsigma^j_{z_j} - \varsigma^i_{z_i+1} + M(1 - \delta^{ij}_{z_i z_j}) &\geq 0 \\ \varsigma^i_{z_i} - \varsigma^j_{z_j+1} + M\delta^{ij}_{z_i z_j} &\geq 0 \\ \delta^{ij}_{z_i z_j} &\in \{0, 1\} \end{aligned} \tag{3.7}$$

$$\begin{aligned} \delta^{ij}_{z_i z_j} = 1 \qquad & \text{if robot } i \text{ goes first along segment } z_i \\ = 0 \qquad & \text{if robot } j \text{ goes first along segment } z_j, \end{aligned}$$

In [56], *Derenick et al.* use conditions on Euclidean Distance Matrix (EDM) [120] to model collision avoidance constraints.

3.2.6 Communication Constraints $\Theta(\zeta, \sigma) \leq 0$

Communication requirements are mostly specified in terms of connectivity between the vehicles. While [13] assumes that there is a centralized communication hub that takes care of all communication needs, [56] explicitly deal with these communication requirements. In [56], *Derenick et al.* use constraints on the Laplacian matrix of

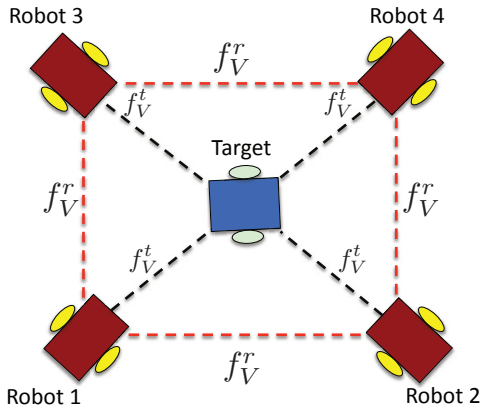

Figure 3.3: A visibility graph with four robots tracking a target. f_V^r is the inter-robot link weight and f_V^t is the robot-target link weight

a communication graph to express communication connectivity requirements. Given a set of nodes $\mathcal{V}_n$ and a set of edges $\mathcal{E}_m$, let $G(\mathcal{V}_n, \mathcal{E}_m)$ be a weighted graph with n vertices and m edges. The Laplacian $L(G)$ of G is a n-dimensional square matrix whose entries are denoted by

$$[L(G)]_{\psi\alpha} = \begin{cases} -w_{\psi\alpha} & \psi \neq \alpha \\ \sum\limits_{\psi \neq k} w_{\psi k} & \psi = \alpha, \end{cases}$$

where, $w_{\psi\alpha}$ is the weight associated with the edge between vertices ψ and α. There are certain important properties of $L(G)$ that make it particularly useful for several applications. Specifically, the communication connectivity constraint states that the communication connectivity graph should be connected at all times and hence the transformation $P_N^T L P_N$ should be positive definite, where P is a n x $(n-1)$ matrix [121].

3.2.7 Other Constraints $\mathcal{O}(\zeta, \sigma) \leq 0$

In [56], *Derenick et al.* deal with visibility/ tracking constraints that require the group of robots to be within a certain Euclidean distance of a target of interest at all times. These visibility/tracking constraints are expressed using a visibility graph approach (Figure. 3.3). The link weights of this graph are functions of inter-robot and robot-target Euclidean distances. By minimizing the negative of the second smallest eigenvalue of the Laplacian of this visibility graph, the authors guarantee tracking of the target by at least one robot at all times. Of importance in this exposition is the key result from spectral graph theory that states that the second smallest eigenvalue of $L(G)$, λ_2, is a measure of the connectivity of G. $\lambda_2 > 0$ is a necessary and sufficient condition to guarantee the connectivity of G. The further away the value from 0, the more connected the graph [121].

3.3 Observations and Design Considerations

We list several observations that are made based on the discussion in the previous section as well as other works from the literature.

3.3.1 Generality

1. **Concepts/results from other disciplines**:

   The framework is general enough to incorporate concepts and results from other disciplines. In [13], the mathematical program is embedded in a decentralized decision-making algorithm. In [19], the authors make use of results from optimal control theory to get bounds on the decision variables, while in [56], the authors use results from spectral graph theory to express visibility and communication constraints. In recent work [105], *Garcia et al.* recast an optimal control problem for multiple spacecraft maneuvering using a two stage planning

process - the first stage uses an augmented Rapid-exploring Random Tree algorithm [115]. The output of the first stage acts as an initial guess to second stage NLP formulations.

2. **Discretization of Space and/or Time**:

   This framework is general enough to allow discretization of space and/or time. The studies [13] and [56] are examples of continuous space and discrete time formulations. In [19], there is an example of discrete space, continuous time formulation. Each robot's path is divided into segments, and all constraints are indexed over the set of segments. This allowed the authors to choose the segment traversal times as continuous decision variables and obtain upper and lower bounds on the segment traversal times by using results from optimal control theory [118].

3. **Path Primitives**:

   This framework is general enough to allow any path primitives such as straight lines and circular arcs [19,68,69], quintic curves and polar splines [76–79], cubic spirals [80], and intrinsic splines [81] among others to be used.

4. **Centralized/Decentralized Decision Making**:

   The framework allows for centralized decision making schemes as seen in case of [19], and [56] or decentralized decision making schemes, wherein multiple decision makers independently solve their optimization problems [13].

5. **Receding Horizon/Model Predictive Control**:

   If computational resources are limited and/or complete information is unavailable, instead of solving a problem for all time steps for all vehicles, one can use

a receding horizon/model predictive approach to solve similar problems over multiple time steps [13, 114].

3.3.2 Advantages accorded by MP

1. **Modeling Environments and Numerical Solvers**:

   Modeling environments such as `AMPL` [102], `GAMS` [103], and `YALMIP` [104] can be used for rapid development of models for testing/simulation purposes. *Schouwenaars et al.* in [13] implement their entire framework using `AMPL`. Many of these tools heavily preprocess the problem before proceeding to solving them thereby speeding up the solution process tremendously. `CPLEX` [88] is one such commercial solver that has been successfully used to solve problems practical MVMP problems in real-time [106, 107]. Furthermore, in recent work, highly efficient code has been written to implement these algorithm in real time [116, 117].

2. **Handling Non-Convexity**:

   As mentioned in Chapter 2, when the MVMP problem has convex objective functions and convex feasible region, polynomial time algorithms can be used to obtain globally optimal solutions. Existing Newton's method based algorithms for nonlinear non-convex problems only guarantee local optimality. It is difficult to express all MVMP problems in a convex form. Constraints such as collision avoidance are inherently non-convex.

   Due to the fact that certain nonlinear constraints add non-convexity to the problem, many models are constructed by linearizing/approximating the nonlinearities and encoding non-convexities using integer variables. This approach enables the use of powerful integer programming solvers. This can be seen in [13], where the authors develop polygonal approximations of a circle. Recent advances in MINLP solution techniques can be leveraged to solve MVMP

variants without the need to linearize/approximate nonlinear, non-convex constraints [87]. Under certain conditions, lack of convexity due to discrete variables can be handled using branch and bound and cutting plan algorithms for MILP, and branch and bound and outer approximation algorithms for MINLP [87].

4. Centralized Multi-Vehicle Path Coordination under Communication Connectivity Constraints

The coordination of the motion of a number of robots (say n) in a shared workspace so that they avoid collisions is known as the *multiple robot path coordination problem* [1, 54]. We study this problem under communication connectivity constraints. Specifically we study the following problem:

Given a group of path-constrained vehicle robots that have fixed and known initial and goal location, generate time-optimal velocity profiles that satisfy kinematic, dynamic, collision avoidance and communication connectivity constraints.

The fixed paths of the robots are represented by piecewise cubic spline curves. The feasibility criteria for trajectories require that the robots' kinematic and dynamic constraints be satisfied, along with avoiding collisions and obeying the communication constraints. The communication constraints require the robots to remain in communication range of a minimum number of co-travelers. Additional constraints that require the communication network graph be connected are developed and enforced.

As mentioned earlier in Chapter 2, multi-vehicle path coordination is a relatively untouched area. We have developed centralized and decentralized solution approaches for this problem.

The following assumptions are made in this exposition.

- Since this study deals with higher level velocity planning, we assume that for each vehicle, there exists a perfect lower level control to achieve the planned position and heading angle without any time delay.

- There is no slipping in the motion of the robots. This is a valid assumption since we enforce kinematic (non-holonomic constraints, continuous first and second

derivatives of the spline paths) and dynamic constraints (upper bounds on the velocities and accelerations) which disallow such slippage.

4.1 Practical Applications

Examples of situations that may be represented by the problem of interest include motion planning of unmanned vehicles in an urban environment, and underwater marine vehicles that rely on acoustic systems for communication [122]. One of the examples that has validated the relevance of this problem is the DARPA Urban Challenge that was held in 2007 and featured close to 11 vehicles driving along a 55 mile network of fixed roads and lanes in an urban environment. The vehicles were required to obey all California traffic laws for driving maneuvers. Maneuvers involved merging into traffic, navigating traffic circles, and safely passing through intersections [123].

4.2 Model Elements

There are three basic elements of this model that capture the details of robot architecture and motion, fixed paths, and inter-robot communication.

4.2.1 Discrete Time Formulation

We formulate this scenario as a discrete time problem with the parameter t representing steps in time. $T_{\max}$ is the time taken by the last robot to reach its end point. At $t = T_{\max}$ the mission is complete. If a robot reaches the goal point before $T_{\max}$, it continues to stay there until the mission is over. However, if required, it can still communicate with other robots.

4.2.2 Robot Architecture

Consider a two-wheeled differential drive mobile robot as shown in Figure 2.3 in Section 2.3.1. The robot moves in a global (X, Y) Cartesian co-ordinate plane and is represented by the following kinematic model with associated non-holonomic constraint (that disallows the robot from sliding sideways).

$$\dot{x} = scos(\theta) \tag{4.1}$$

$$\dot{y} = ssin(\theta) \tag{4.2}$$

$$\dot{\theta} = \omega \tag{4.3}$$

$$\dot{x}sin(\theta) - \dot{y}cos(\theta) = 0 \tag{4.4}$$

Here s and ω are the linear and angular speeds of the robot respectively. x, y and θ are the coordinates of the robot with respect to the global (X, Y) coordinate system.

Consider a group of n such mobile robots. Each robot $i = 1, \ldots, n$ is represented by a common mathematical model $(4.1) - (4.3)$ with associated non-holonomic constraint (4.4) and has a fixed path p^i to follow, with a given start (origin) point o^i and a given end (goal) point e^i. P is the set of all the fixed paths of each robot. $p^i \in P, \forall i = 1, \ldots, n$. O is the set of all start (origin) points. $o^i \in O, \forall i = i = 1, \ldots, n$. E is the set of all end points. $e^i \in E, \forall i = i = 1, \ldots, n$. The Euclidean distance between two robots i and j is denoted by d^{ij}. The robots are required to maintain a minimum safe distance d_{safe} in order to avoid collisions with each other. The distance between the current location and the goal point for robot i is given by d^i_{goal}. s^i denotes the speed of the robot i along its fixed path.

4.2.3 Robot Paths

Each robot follows a fixed path represented by a two dimensional piecewise cubic spline curve. The curve is obtained by combining two one dimensional piecewise cubic splines $px(u)$, and $py(u)$, where the parameter u is arc length along the curve and $px(u)$ and $py(u)$ are given by (4.5) and (4.6).

$$px(u) = \varrho_{x0} + \varrho_{x1}u + \varrho_{x2}u^2 + \varrho_{x3}u^3 \tag{4.5}$$

$$py(u) = \varrho_{y0} + \varrho_{y1}u + \varrho_{y2}u^2 + \varrho_{y3}u^3 \tag{4.6}$$

where $\varrho_{x0}, \ldots, \varrho_{x3}$ and $\varrho_{y0}, \ldots, \varrho_{y3}$ are the coefficients of $px(u)$ and $py(u)$ individual splines respectively.

The curvature κ at any point along a two dimensional spline curve is given by

$$\kappa(u) = \frac{px'(u)py''(u) - py'(u)px''(u)}{(px'(u)^2 + py'(u)^2)^{3/2}} \tag{4.7}$$

For each robot i, the angular speed can be calculated as

$$\omega^i(u) = s^i(u)\kappa^i(u) \tag{4.8}$$

These piecewise cubic splines have continuous first derivatives (slope) and second derivatives (curvature) along the curve. This property makes the path kinematically feasible for the two-wheeled differential drive robots. Furthermore, upper and lower bounds on the speed, acceleration and angular speed (turning rate) are enforced, thereby taking into account the robot dynamics. The paths represented by the two dimensional piecewise cubic splines along with the constraints on the speed, accelerations and turn rates result in a kinodynamically feasible trajectory. For a detailed discussion on spline curve design and analysis, readers are referred to [124] and its references. Figures 4.1 and 4.2 show how the spline curve paths are constructed.

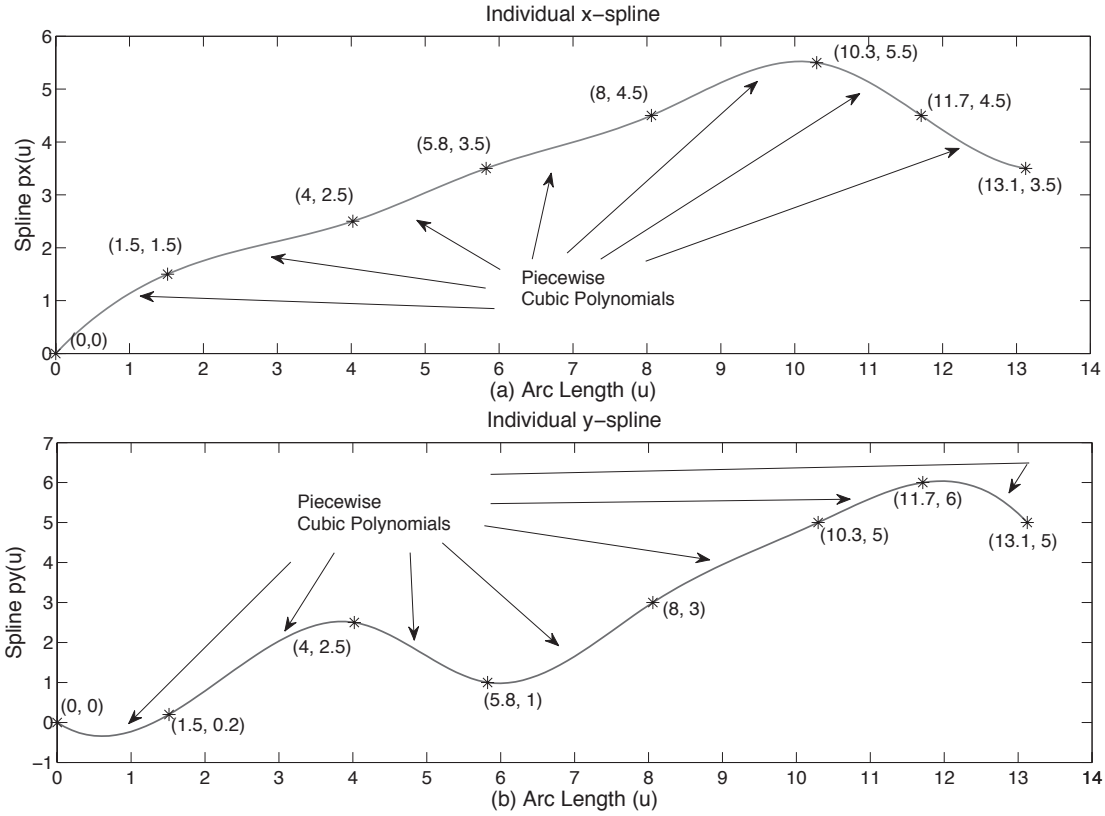

Figure 4.1: $px(u)$ and $py(u)$ are individual cubic splines constructed where u is the arc length along the curve.

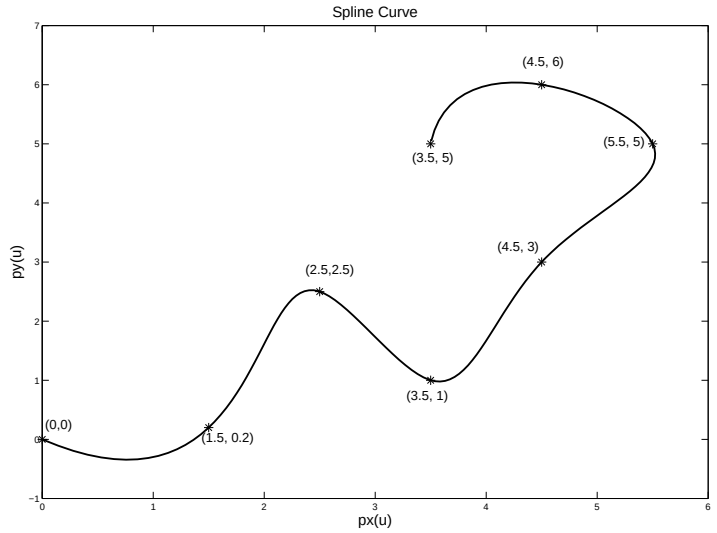

Figure 4.2: The spline curve is obtained by combining $px(u)$ and $py(u)$.

Other path primitives that result in twice continuously differentiable functions in our constraints such as quintic curves, polar splines, cubic spirals can be easily accommodated using this framework.

4.2.4 Communication Model

Each robot is equipped with a wireless transceiver. The communication constraint requires that at all times, every robot is in communication range of at least n_{conn} other robots, where n_{conn} varies between 0 to $n-1$. The Signal to Noise Ratio (SNR) experienced by the receiver robot is calculated to determine whether or not the communication constraint is satisfied or not. If the SNR experienced by a receiver node placed on a robot is above a predefined threshold η_c, the two robots are considered to be in one-hop communication range of each other.

Consider two robots i and j that try to communicate with each other at a given point in time. The Euclidean distance between them is denoted by d^{ij}. The signal transmission power of the wireless transceiver placed on the transmitter robot is denoted by P_t^i. The received signal power of the wireless transceiver placed on the receiver robot is denoted by P_r^j.

As described in Chapter 2, the received power P_r^j can be described using (2.25) as (4.9)

$$P_r^j(d^{ij})[dBm] = P_t^i[dBm] - PL^{ij}(d^{ij})[dB] \tag{4.9}$$

where $PL^{ij}(d^{ij})[dB]$ is the path-loss experienced by the transmitter and receiver robot pair i-j, and is expressed using (2.24) as (4.10)

$$PL^{ij}(d^{ij})[dB] = \overline{PL^{ij}}(d_0) + \Upsilon_\upsilon = \overline{PL^{ij}}(d_0) + 10\alpha \log\left(\frac{d^{ij}}{d_0}\right) + \Upsilon_\upsilon \tag{4.10}$$

The Signal to Noise Ratio (SNR) experienced by the receiver robot j is calculated using the relationship

$$\mathrm{SNR}^j = P_r^j / N, \tag{4.11}$$

where $N = kTBF$ is the thermal noise experienced by receiver j.

SNR levels are used to determine whether the robots are in communication range of each other. If the outage probability of communication $\mathbb{P}_0^{ij}$ for transmitter and receiver robot pair i-j is below a threshold η_ϵ, the the robots are considered to be in one-hop communication range of each other.

$$\begin{aligned} &\mathbb{P}_0^{ij} \leq \eta_\epsilon \\ =&\mathbb{P}(\mathrm{SNR}^{ij} < \eta_{snr}) \leq \eta_\epsilon \\ =&\mathbb{P}(P_r^j(d^{ij}) < \eta_c) \leq \eta_\epsilon \end{aligned} \tag{4.12}$$

4.3 Centralized Formulations

In all the centralized formulations, we seek to minimize $T_{\max}$, the time of arrival of the last arriving robot, while each robot is in one-hop communication range of at least n_{conn} other robots at all time steps, and satisfying the kinodynamic and collision avoidance constraints. The optimization is performed over the speeds of the robots along the specified paths. Since $T_{\max}$ is not known a priori, we pick a sufficiently large number of time steps T ($T_{\max} \leq T$) in our model so that it will yield a feasible solution.

4.3.1 Primary MINLP Formulation

For a given value of n_{conn} between 0 to $n-1$, the equations (4.13)-(4.26) model the optimization problem at hand. The formulation is a MINLP problem with inter-robot communication constraints and $T_{\max}$ being specified using binary variables. The rest of the variables in the formulation are continuous variables.

$$\text{minimize} \quad \sigma_a T_{\max} + \sigma_b \sum_{i,t} d^i_{goal}(t) \tag{4.13}$$

$$\forall i, j \in \{1 \dots n\}, j \neq i, \forall t \in \{1 \dots T\},$$

$$\text{subject to} \quad (x^i(0), y^i(0)) = o^i \tag{4.14}$$

$$(x^i(T), y^i(T)) = e^i \tag{4.15}$$

$$u^i(0) = 0 \tag{4.16}$$

$$u^i(t) = u^i(t-1) + s^i(t)\Delta t \tag{4.17}$$

$$(x^i(t), y^i(t)) = ps^i(u^i(t)) \tag{4.18}$$

$$s_{min} \leq s^i(t) \leq s_{max} \tag{4.19}$$

$$a_{min} \leq a^i(t) \leq a_{max} \tag{4.20}$$

$$d^{ij}(t) \geq d_{safe} \tag{4.21}$$

$$d^i_{goal}(t) = U^i - u^i(t) \tag{4.22}$$

$$A^i(t) = \begin{cases} 0 & \text{if } d^i_{goal}(t) \neq 0 \\ 1 & \text{if } d^i_{goal}(t) = 0 \end{cases} \tag{4.23}$$

$$T_{\max} = \max_{i=1,\dots,n} \left(\sum_{t=0}^{T} \left(1 - A^i(t)\right) \right) \tag{4.24}$$

$$\sum_{j:j\neq i} C^{ij}(t) \geq n_{conn} \tag{4.25}$$

$$C^{ij}(t) = \begin{cases} 1 & \text{if } \mathbb{P}(SNR^{ij}(t) < \eta_{snr}) \leq \eta_\epsilon \\ 0 & \text{if } \mathbb{P}(SNR^{ij}(t) < \eta_{snr}) > \eta_\epsilon \end{cases} \tag{4.26}$$

A. **Decision Variables**

In (4.13)-(4.26), the main decision variables are the speeds, $s^i(t)$, for vehicle $i = 1 \dots n$ at time $t = 1 \dots T$. The values of the remaining variables are dependent on the speeds, as described in the following sections on the problem constraints.

B. **Objective Function**

Equation (4.13) represents the objective function to be minimized. The first term of the objective function is $T_{\max}$ which represents the time taken by the last robot to reach its goal point. σ_a and σ_b are penalty parameters. By using σ_b, the second term forces the robots to minimize the distance between their current location and the goal position and prevents the robots from stalling. Constraint (4.22) defines the distance to goal $d^i_{goal}(t)$ for each robot $i = 1 \dots n$ at time-step $t = 1 \dots T$ as the difference between its path length U^i and the total arc length travelled $u^i(t)$.

C. **Path (Kinematic) Constraint**

Constraints (4.14)-(4.18) define the path of each robot. Constraints (4.14) and (4.15) form the set of boundary requirements that each robot $i = 1 \dots n$ has to start at a designated start point o^i and finish at a designated end point e^i at the end of the planning horizon. Constraint (4.16) initializes the arc length travelled $u^i(t)$ to zero value. Constraint (4.17) increments the arc length $u^i(t)$ at each time $t = 1 \dots T$ step based on the speed of the robot $i = 1 \dots n$ ($\Delta t = 1$). Constraint (4.18) ensures that the robots follow their respective paths as defined by the cubic splines. The function $ps^i(u^i(t))$ denotes the location of robot $i = 1 \dots n$ at time step $t = 1 \dots T$ after traveling an arc length of u along the piecewise cubic spline curves.

D. **Speed and Acceleration (Dynamic) Constraint**

Constraints (4.19) and (4.20) are dynamic constraints and ensure that the speed $s^i(t)$ (and hence, angular speed) and the acceleration $a^i(t)$ respectively are bounded above and below for each robot $i = 1 \dots n$ at each time step $t = 1 \dots T$. These constraints are determined by the capabilities of the robot. In general, solving an optimal path planning problem consists of finding a set of feasible pairs of linear and angular velocities that minimize a given cost function. Here, we assume that the maximum curvature of the path is within the achievable bounds of the angular velocity and radial acceleration of the robots, and so the angular speed corresponding to the optimal speed will always be achievable. Hence the overall solution will be feasible. In absence of such an assumption, by adding a constraint on the angular velocity and radial acceleration in the model, the feasible set of solutions for any given path with an associated curvature can be determined.

E. **Collision Avoidance Constraint**

The constraint (4.21) ensures that there is a sufficiently large distance d_{safe} between each pair of robots $i, j = 1 \dots n, i \neq j$ to avoid a collision. This is a non-convex constraint that an inherent part of any motion planning problem. As mentioned in Section 2.4.2, dealing with non-convexity is a challenging task, and the solution algorithms used in this thesis only guarantee local optimality for nonlinear non-convex problems.

F. **Definition of $T_{\max}$**

As defined by constraint (4.23), $A^i(t)$ is a binary variable used to indicate whether the robot $i = 1 \dots n$ has reached its destination or not. If $A^i(t) = 1$ for all (i, t) with $d^i_{goal}(t) = 0$, the total amount of time it takes a robot i to reach

its destination can be calculated using (4.27)

$$\sum_{t=0}^{T} \left(1 - A^i(t)\right) \tag{4.27}$$

$T_{\max}$ as defined by constraint (4.24) is the maximum time taken by the last arriving robot to reach its destination.

G. **Communication Connectivity Constraint**

Constraints (4.25) and (4.26) state that vehicle $i = 1 \dots n$ should be in one-hop communication range of at least n_{conn} robot vehicles. This means that, for at least n_{conn} values of $j = 1, \dots, n$, $j \neq i$, the condition $\mathbb{P}(SNR^{ij}(t) \leq \eta_c) \leq \eta_\epsilon$ should be satisfied. The remaining robot vehicles may or may not be in one-hop communication range of i. In order to express this requirement, we introduce a binary variable $C^{ij}(t)$ that indicates whether the two robots are in one-hop communication range of each other or not. $C^{ij}(t) = 1$ if for the robots i and j if they are in one-hop communication range of each other at time t i.e. $\mathbb{P}(SNR^{ij}(t) \leq \eta_{snr}) \leq \eta_\epsilon$ and $C^{ij}(t) = 0$ if $\mathbb{P}(SNR^{ij}(t) \leq \eta_{snr}) > \eta_\epsilon$

4.3.2 Robust MINLP Reformulation

We take a closer look at the communication constraints of the primary formulation presented in Section 4.3.1. The outage probability for the transmitter receiver robot pair i-j is be given by (4.29) and (4.28).

$$\mathbb{P}_0^{ij} = \mathbb{P}(P_r^j(d^{ij}) < \eta_c) = Q\left(\frac{\overline{P_r(d^{ij})} - \eta_c}{\upsilon}\right) \tag{4.28}$$

and

$$1 - \mathbb{P}_0^{ij} = \mathbb{P}(P_r^j(d^{ij}) > \eta_c) = Q\left(\frac{\eta_c - \overline{P_r(d^{ij})}}{\upsilon}\right) \tag{4.29}$$

where $Q(\cdot)$ is the Q-function represented by (4.30)

$$Q(z) = \frac{1}{\sqrt{2\pi}} \int_{z}^{\infty} e^{-\frac{x^2}{2}} dx \tag{4.30}$$

For a given outage probability threshold for the transmitter receiver robot pair i-j, η_ϵ, the condition $\mathbb{P}_0^{ij} \leq \eta_\epsilon$ leads to (4.31).

$$\begin{aligned}
&\mathbb{P}_0^{ij} \leq \eta_\epsilon \\
\Longleftrightarrow &\mathbb{P}(SNR^{ij}(t) < \eta_{snr}) \leq \eta_\epsilon \\
\Longleftrightarrow &\mathbb{P}(P_r^j(d^{ij}) < \eta_c) \leq \eta_\epsilon \\
\Longleftrightarrow &Q\left(\frac{\overline{P_r(d^{ij})} - \eta_c}{\upsilon}\right) \leq \eta_\epsilon \\
\Longleftrightarrow &\frac{\overline{P_r(d^{ij})} - \eta_c}{\upsilon} \leq Q^{-1}(\eta_\epsilon) \\
\Longleftrightarrow &\overline{P_r(d^{ij})} \leq \eta_c + \upsilon Q^{-1}(\eta_\epsilon) \\
\Longleftrightarrow &P_t[dBm] - \overline{PL}(d_0) - 10\alpha \log \frac{d^{ij}}{d_0} \leq \eta_c + \upsilon Q^{-1}(\eta_\epsilon) \\
\Longleftrightarrow &-10\alpha \log(d^{ij}) \leq \eta_c + \upsilon Q^{-1}(\eta_\epsilon) - P_t[dBm] + \overline{PL}(d_0) - 10\alpha \log(d_0) \\
\Longleftrightarrow &\log(d^{ij}) \leq \frac{\eta_c + \upsilon Q^{-1}(\eta_\epsilon) - P_t[dBm] + \overline{PL}(d_0) - 10\alpha \log(d_0)}{-10\alpha} \\
\Longleftrightarrow &d^{ij} \leq 10\hat{} \left(\frac{\eta_c + \upsilon Q^{-1}(\eta_\epsilon) - P_t[dBm] + \overline{PL}(d_0) - 10\alpha \log(d_0)}{-10\alpha}\right)
\end{aligned} \tag{4.31}$$

Thus, the condition $\mathbb{P}_0^{ij} \leq \eta_\epsilon$ leads to the condition $d^{ij} \leq \eta_{rd}$, where η_{rd} is given by right-hand side of the inequality (4.31). Then the communication constraints (4.25) and (4.26) in the primary formulation presented in Section 4.3.1 can be expressed by (4.32) and (4.33)

$$\sum_{j:j\neq i} C^{ij}(t) \geq n_{conn} \tag{4.32}$$

$$C^{ij}(t) = \begin{cases} 1 & \text{if } d^{ij} \leq \eta_{rd} \\ 0 & \text{if } d^{ij} > \eta_{rd} \end{cases} \tag{4.33}$$

A. **Communication constraint reformulation**

The disjunctive one-hop communication constraints (4.32) and (4.33) are reformulated as Big-M constraints given by (4.34)-(4.36) [84]. For this reformulation, we introduce a constant M_1 and formulate constraint (4.35). In case $d^{ij}(t) \leq \eta_d$, $C^{ij}(t)$ can assume a value of either 0 or 1. But the constraint (4.36) forces at least n_{conn} of them to be set to 1. This means that only n_{conn} of the C^{ij} variables, and not necessarily all, are guaranteed to have the correct value $C^{ij} = 1$. It is easily seen that if $C^{ij}(t) = 0$, then the constraint will be trivially satisfied for a sufficiently large M_1.

$$C^{ij}(t) \in \{0, 1\} \tag{4.34}$$

$$d^{ij}(t) \leq M_1(1 - C^{ij}(t)) + \eta_d \tag{4.35}$$

$$\sum_{j:j\neq i} C^{ij}(t) \geq n_{conn}, \quad C^{ij}(t) \in \{0, 1\} \tag{4.36}$$

Note: In our implementation, we used an equivalent form of constraint (4.35) as (4.37).

$$\frac{(d^{ij}(t))^2}{(M_1(1 - C^{ij}(t)) + \eta_d)} \leq (M_1(1 - C^{ij}(t)) + \eta_d) \tag{4.37}$$

This form is chosen in order to avoid the nondifferentiability of a Euclidean distance calculation within the nonlinear solver. The nondifferentiability is not going to occur at the optimal solution due to the collision avoidance constraint keeping d^{ij} sufficiently large, but during the initial iterations of the MILANO solver, it may cause numerical difficulties. The reformulation removes the potential of such an occurrence and provides numerical stability.

B. **Reformulation of $T_{\max}$ definition**

The definition of $T_{\max}$ in (4.23)-(4.24) is reformulated as (4.38)-(4.40).

$$A^i(t) \in \{0, 1\} \tag{4.38}$$

$$d^i_{goal}(t) - M_2(1 - A^i(t)) \leq 0 \tag{4.39}$$

$$T_{\max} \geq \left(\sum_{t=0}^{T} \big(1 - A^i(t)\big) \right), \forall i \in \{1 \dots n\} \tag{4.40}$$

If $A^i(t) = 1$ for all (i, t) with $d^i_{goal}(t) = 0$, the total amount of time it takes a robot i to reach its destination can be calculated using (4.41).

$$\sum_{t=0}^{T} \big(1 - A^i(t)\big) \tag{4.41}$$

When $d^i_{goal}(t) = 0$, $A^i(t)$ is free to be either 0 or 1. However, constraint (4.40) specifies $T_{\max}$ as an upper bound for the sum (4.41), and equation (4.13) minimizes $T_{\max}$. Therefore at the optimal solution, $A^i(t) = 1$ when robot i is at its destination at time t and that (4.40) will hold with equality.

When $A^i(t) = 1$ in constraint (4.39), $d^i_{goal}(t) \leq 0$, which along with fact that

distance to goal is always non-negative ($d^i_{goal}(t) \geq 0$) leads to the condition $d^i_{goal}(t) = 0$. If $A^i(t) = 0$, then $d^i_{goal}(t) \leq 0$, a condition that will be trivially satisfied for a value of M_2 that is sufficiently bigger than the path-lengths of the robots.

The reformulated robust MINLP formulation $\mathcal{O}_{cen}$ is given by (4.42)-(4.57).

$$\text{minimize} \quad \sigma_a T_{\max} + \sigma_b \sum_{i,t} d^i_{goal}(t) \tag{4.42}$$

$$\forall i, j \in \{1 \dots n\}, j \neq i, \forall t \in \{1 \dots T\}$$

$$\text{subject to} \quad (x^i(0), y^i(0)) = o^i \tag{4.43}$$

$$(x^i(T), y^i(T)) = e^i \tag{4.44}$$

$$u^i(0) = 0 \tag{4.45}$$

$$u^i(t) = u^i(t-1) + s^i(t)\Delta t \tag{4.46}$$

$$(x^i(t), y^i(t)) = ps^i(u^i(t)) \tag{4.47}$$

$$s_{min} \leq s^i(t) \leq s_{max} \tag{4.48}$$

$$a_{min} \leq a^i(t) \leq a_{max} \tag{4.49}$$

$$d^{ij}(t) \geq d_{safe} \tag{4.50}$$

$$d^i_{goal}(t) = U^i - u^i(t) \tag{4.51}$$

$$d^i_{goal}(t) - M_2(1 - A^i(t)) \leq 0 \tag{4.52}$$

$$T_{\max} \geq \left(\sum_{t=0}^{T} \left(1 - A^i(t)\right) \right), \forall i \in \{1 \dots n\} \tag{4.53}$$

$$A^i(t) \in \{0, 1\} \tag{4.54}$$

$$d^{ij}(t) \leq M_1(1 - C^{ij}(t)) + \eta_d \tag{4.55}$$

$$\sum_{j:j \neq i} C^{ij}(t) \geq n_{conn} \tag{4.56}$$

$$C^{ij}(t) \in \{0, 1\} \tag{4.57}$$

4.3.3 Connectivity Partition Elimination Constraint

The one-hop communication connectivity constraints will not necessarily prevent partition of the network to non-communicating subgroups. A constraint of the form

$$\sum_{i \in I, j \notin I} C^{ij}(t) \geq 1 \tag{4.58}$$

for each subgroup I of size $n_{conn}+1, \cdots, n-1$ at each time period would ensure that there exists at least one connection between each subgroup thereby resulting in connectivity throughout the network. We will refer to (4.58) as the *"partition elimination"* (P.E.) constraints.

The number of partition elimination constraints n_{part} to be added to the problem expressed by (4.42)-(4.57) for all possible partitions is given by

$$n_{part} = \begin{cases} \sum_{i=k+1}^{\lfloor n/2 \rfloor} \binom{n}{i} & n \text{ is odd} \\ \\ \sum_{i=k+1}^{\lfloor n/2 \rfloor} \binom{n}{i} - \frac{1}{2}\binom{n}{\frac{n}{2}} & n \text{ is even} \end{cases} \tag{4.59}$$

Adding partition elimination constraints for all possible partitions for each time period to the problem expressed by (4.42)-(4.57) would increase its size exponentially. Instead, we apply Algorithm 1 (also referred to as the P.E. algorithm), which solves the problem without these constraints first and then detects any partitions in the solution. The P.E. algorithm is a form of breadth first search algorithm that searches the entire communication connectivity graph to detect partitions. For each partition detected, we add one constraint of the form (4.58) to (4.42)-(4.57). The updated problem is re-solved and the process is continued until no more partitions are detected. In our numerical simulations, we found that this greatly reduces the size of

the problem. By using the partition elimination algorithm at each iteration, we solve a relaxation of the actual problem.

Algorithm 1 Partition Elimination (P.E.)

Require: n robots with fixed paths
Ensure: Eliminate partitions
  repeat
    Solve the model.
    Let Done = FALSE
    for $t \in \{1, \ldots, T\}$ **do**
      Let $I = \{1\}$.
      for $i \in I, j \notin I$ **do**
        if $C^{ij}(t) > 0$ **then**
          $I = I \cup \{j\}$
        end if
      end for
      if $I = \{1, \ldots, n\}$ **then**
        Done = TRUE
      else
        add the constraint $\sum_{i \in I, j \notin I} C^{ij}(t) \geq 1$ to the model
      end if
    end for
  until Done = FALSE

4.3.4 NLP Reformulation

The robust MINLP reformulation (4.42)-(4.57) that included the P.E. constraints (4.58) was reformulated to remove the integer variables. The resulting reformulation is a NLP that was implemented in `AMPL` and solved using `LOQO`. This NLP model and associate numerical results have been described in Appendix A.

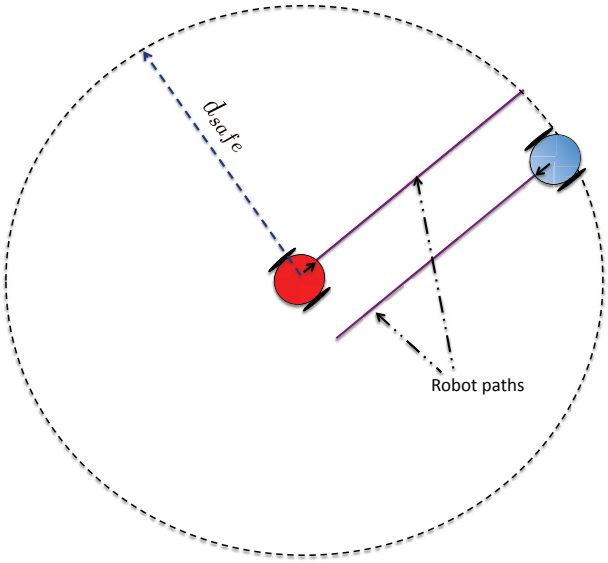

Figure 4.3: The radius of the dotted circle here represents d_{safe}. The two robots are at their starting points. It is clear that with this d_{safe} value, the problem is infeasible.

4.4 Feasibility Considerations

We identify and formalize several conditions that will result in feasibility of $\mathcal{O}_{cen}$ as defined by (4.42)-(4.57). We start by formalizing the notion of feasible configurations and noting the Lemma 4.4.3 which states that the communication constraints are the only source of infeasibility for $\mathcal{O}_{cen}$.

Lemma 4.4.1. *The collision avoidance distance d_{safe} should satisfy the following property to not cause infeasibility in $\mathcal{O}_{cen}$.*

$$d_{safe} \ll min(max\ d^{ij}(t))\forall i, j = 1, 2, \ldots, n, \forall t = 1, 2, \ldots, T, j \neq i \tag{4.60}$$

Proof. If condition (4.60) is not satisfied, then there can be a case as shown in Figure 4.3. It is trivial to see that (4.60) is required as it enforces a conservative limit on the value of d_{safe} such that, at all times, it is sufficiently less than the maximum separation between any two robots. □

Definition 4.4.2. (Feasible configuration) *Define a feasible configuration $\mathcal{F}(t)$ at discrete time-step t as the set of locations $(x^i(t), y^i(t))$ along the path p^i for each robot i such that no robots collide with each other. i.e.*

$$\mathcal{F}(t) = \{(x^i(t), y^i(t)), \forall i, j \in 1, \ldots, n, \forall t \in 1, \ldots, T, j \neq i |\ d^{ij} \geq d_{safe}\} \tag{4.61}$$

d_{safe} satisfies (4.60).

Lemma 4.4.3. *Given a set of n robots with fixed paths whose start (origin) and end (destination) points are feasible configurations, in absence of the communication constraints, the mathematical program $\mathcal{O}_{cen}$ will always be feasible if the condition* (4.60) *is satisfied.*

Proof. The communication connectivity constraints restrict the robots to move such that they remain in communication range of each other. In absence of such constraints, a trivial feasible solution for $\mathcal{O}_{cen}$ is always available wherein the robots move one after the other such that only one robot moves at a time and the other robots wait till the robot in transit reaches its destination. □

Next we introduce definitions that will aid us in developing notions of feasibility with the communication constraints present in $\mathcal{O}_{cen}$.

Definition 4.4.4. (Infeasible Paths) *The set of paths P is defined as infeasible paths if for some path $p^i \in P$ of robot i, there exists a closed interval of arc-lengths $[u^i_{infe\_start}, u^i_{infe\_end}]$ such that $\forall u^i_{infe} \in [u^i_{infe\_start}, u^i_{infe\_end}]$ condition* (4.62) *is satisfied.*

$$\sqrt{(x^i(u^i_{infe}) - x^j(u^j))^2 + (y^i(u^i_{infe}) - y^j(u^j))^2} > \eta_d \tag{4.62}$$

$\forall u^j \in [0, U^j]$, $\forall j \in \mathbb{Y}$. The set $\mathbb{Y} \subset \{1, 2, \ldots, n\}$ with cardinality $|\mathbb{Y}| > n - n_{conn} - 1$.

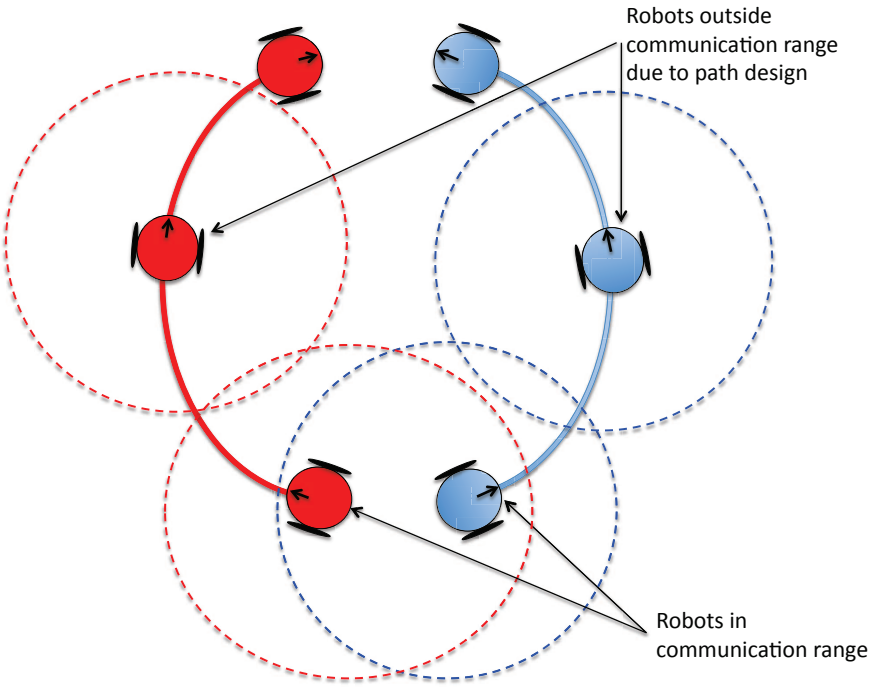

Figure 4.4: The paths shown here are *Infeasible Paths*. The dotted circles indicate the communication range of each robot.

An example of *Infeasible Paths* is described in Figure 4.4. This set is an example of a case where the communication constraints are not satisfied due to path limitations. This condition is sufficient for $\mathcal{O}_{cen}$ to be infeasible. The following definition characterizes another condition that can cause infeasibility.

Definition 4.4.5. (Infeasible Set) *The set of paths P is defined as infeasible set if for some closed interval of arc-lengths $[u^i_{dinfe\_start}, u^i_{dinfe\_end}]$ along the path $p^i \in P$ for robot i, there exists some closed interval of arc-lengths $[u^j_{dinfe\_start}, u^j_{dinfe\_end}]$ along the path $p^j \in P$ for some robot $j \in \mathbb{Y}$, where the set $\mathbb{Y} \subset \{1, 2, \ldots, n\}$ with cardinality $|\mathbb{Y}| > n - n_{conn} - 1$ such that the robot j has to move at infinite speed to satisfy the condition* (4.63)

$$\sqrt{(x^i(u^i_{dinfe}(t)) - x^j(u^j_{dinfe}(t)))^2 + (y^i(u^i_{dinfe}(t)) - y^j(u^j_{dinfe}(t)))^2} \leq \eta_d \tag{4.63}$$

An infeasible set is shown in the Figure 4.5. The top figure shows the initial feasible configuration of five (5) robots with $n_{conn} = 2$. The two dotted circles indicate the communication range of the red and yellow robots, respectively. In this configuration, all robots are in communication with 2 other robots. It is also apparent that these paths are not *Infeasible Paths* as per Definition 4.4.4. However, as shown in the bottom figure, at some point during the transit of the red robot, the blue robot moves outside its communication range violating the communication connectivity constraint of $n_{conn} = 2$ at all time steps. Unless, the blue robot has infinite acceleration so that it can traverse the segment of its path that lies outside the communication range of the red robot instantaneously, the problem is infeasible. This is an example of a scenario where the communication constraints are not satisfied due to dynamic bounds.

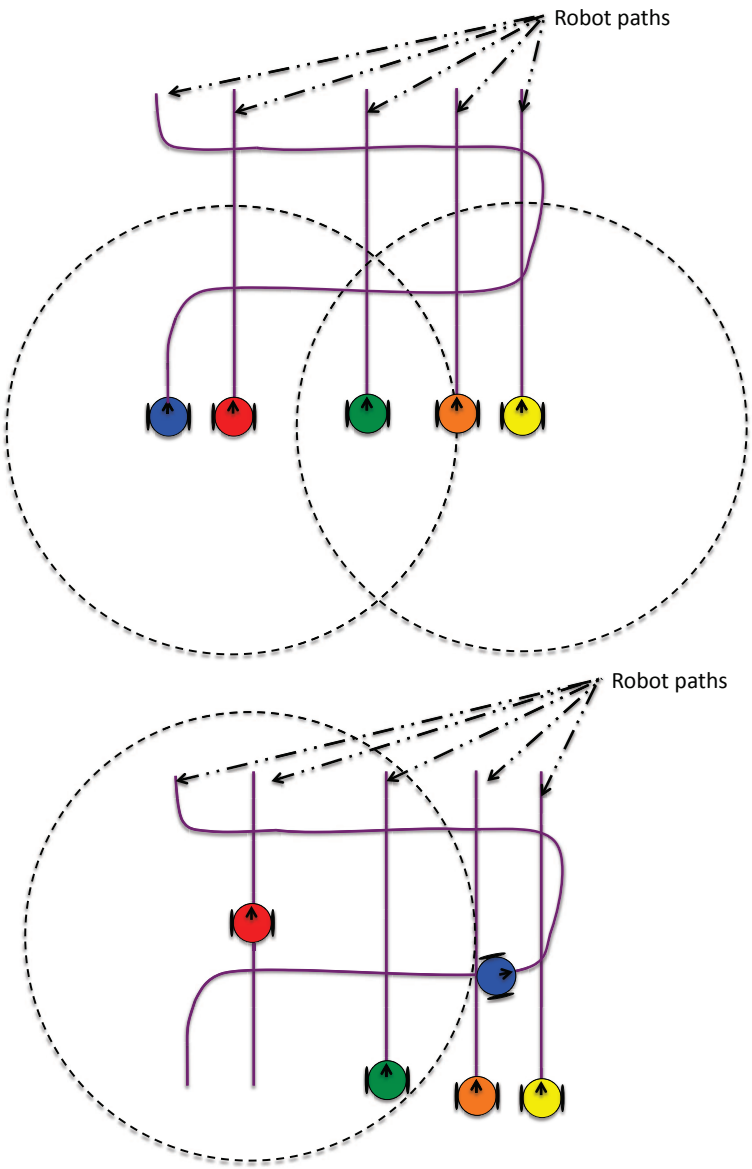

Figure 4.5: The paths shown here form an *Infeasible Set*. The dotted circles indicate the communication range of each robot.

5. Decentralized Multi-Vehicle Path Coordination under Communication Connectivity Constraints

The high computational requirements of the centralized formulation motivate the study of a decentralized alternative. In addition, in many scenarios, new information about the environment becomes available to the robots as they progress, thereby requiring changes in the coordination scheme. In such cases, a decentralized receding horizon planning strategy is particularly attractive as it allows the robots to quickly resolve and communicate their plan with other robots in the group. Also, since we do not know the actual value of the scenario completion time beforehand, using a receding horizon allows to keep the problem size (number of variables) limited.

In this chapter, we present a sequential, decentralized framework to solve the multi-vehicle path coordination problem under communication constraints. Each vehicle robot solves only its own coordination problem, formulated as a Receding Horizon Mixed Integer Nonlinear Programming problem (RH-MINLP) problem subject to constraints on kinematics, dynamics, collision avoidance, and communication connectivity. The sequential nature of the formulation forces each robot to solve its own problem in a predetermined order and exchange trajectory information with other robots. In this way, each robot can take into account the latest trajectory of other robots while planning its own trajectory.

The fixed paths of the robots are represented by piecewise cubic spline curves. The feasibility criteria for trajectories require that the robots' kinematic and dynamic constraints be satisfied, along with the imperatives of avoiding collisions and obeying the communication connectivity constraints. The receding horizon formulation is implemented in `MATLAB`, which is interfaced with the MINLP solver `MILANO` [8]. `MILANO` is a `MATLAB`-based solver for mixed-integer linear and nonlinear programming

problems. It uses a branch-and-bound method for handling integer variables, and an interior-point method for solving the nonlinear relaxations. Source code for `MILANO` has been made available online [125].

The following assumptions are made:

- Fixed path coordination and motion planning is fully decentralized.
- While robots are required to maintain a certain level of communication connectivity with their immediate neighbors, there is a central communication station that allows robots to communicate trajectory data to all the robots in the group and keep individual clocks synchronized.
- Similar to the centralized formulations, we assume that for each vehicle, there exists a perfect lower level control to achieve the planned position and heading angle without any time delay.
- There is no slipping in the motion of the robots.

5.1 Model Elements

5.1.1 Receding Horizon

Similar to the centralized formulations, the parameter t represents discrete steps in time. T_{hor} is the receding horizon time. At each time step t, each robot must calculate its plan for the next T_{hor} time steps, and communicate this plan with other robots in the network. While the robots compute their trajectory points and corresponding input commands for the next T_{hor} time steps, only the first of these solutions is implemented, and the process is repeated at each time step. T_{max} is the time taken by the last arriving robot to reach its end point. At $t = T_{max}$ the scenario is completed. If a robot reaches the goal point before T_{max}, it continues to stay there untill the mission is over. However, if required, it can still communicate with other robots.

Each robot plans its own trajectory at each discrete time step, by taking into account the plans of all other robots. For a given time step t, each robot determines its speed for the next T_{hor} time steps starting time t i.e. $s^i(t), \ldots, s^i(t+T_{hor})$ and implements the first speed $s^i(t)$ out of all these speeds. In this way the plan starting at time step $t+1$ must be computed during time step t. Thus during each time step t, each robot communicates the following information about its plan $\mathcal{P}^i(t)$ to other robots: $\mathcal{P}^i(t) = [\mathbf{p}^i(t+1) \ldots \mathbf{p}^i(t+T_{hor})]$, where $\mathbf{p}^i(t) = (x^i(t), y^i(t))$ is the location of the robot i on its path at time t calculated based on the optimal speed $s^i(t)$.

5.1.2 Decision Ordering

The robots are assigned a pre-determined randomized decision order. The decentralized algorithm presented here sequentially cycles through each robot thereby allowing each robot to solve its planning problem in the order $ord(i)$, $i \in \{1, 2, ..., n\}$. While the general structure of this problem is similar to the centralized problem, in the following sections, we discuss the decision variables and constraints for the sake of clarity.

5.2 Decentralized Formulation

Each robot i solves the optimization problem $\mathcal{O}^i(t)$ indicated by (5.1)-(5.13) in the order $ord(i)$ that it has been assigned.

$$\text{minimize} \quad \sum_{k=t}^{t+T_{hor}} d^i_{goal}(k) \tag{5.1}$$

$$\text{subject to} \quad \forall j \in \{1, 2, \ldots, n\}, j \neq i$$

$$\forall k \in \{t, \ldots, t + T_{hor}\}$$

$$(x^i(0), y^i(0)) = o^i \tag{5.2}$$

$$u^i(0) = 0 \tag{5.3}$$

$$u^i(k) \leq U^i \tag{5.4}$$

$$u^i(k) = u^i(k-1) + s^i(k)\Delta t \tag{5.5}$$

$$(x^i(k), y^i(k)) = ps^i(u^i(k)) \tag{5.6}$$

$$s_{min} \leq s^i(k) \leq s_{max} \tag{5.7}$$

$$a_{min} \leq a^i(k) \leq a_{max} \tag{5.8}$$

$$d^i_{goal}(k) = U^i - u^i(k) \tag{5.9}$$

$$d^{ij}(k) \geq d_{safe} \tag{5.10}$$

$$d^{ij}(k) \leq M(1 - C^{ij}(k)) + \eta_d \tag{5.11}$$

$$\sum_{j:j\neq i} C^{ij}(k) \geq n_{conn} \tag{5.12}$$

$$C^{ij}(k) \in \{0, 1\} \tag{5.13}$$

5.2.1 Decision Variables

In (5.1)-(5.13), the main decision variables are the speeds, $s^i(t), \ldots, s^i(t + T_{hor})$, for robot i at time t. The values of the remaining variables are dependent on the

speeds.

5.2.2 Objective Function

Equation (5.1) represents the objective function to be minimized. This formulation forces the robots to minimize the total distance between their current location and the goal position over the entire receding horizon. Constraint (5.9) defines the distance to goal $d^i_{goal}(k)$ for each robot $i = 1 \dots n$ at time-step k, $\forall k \in \{t, \dots, t+T_{hor}\}$ as the difference between it's path length U^i and the total arc length travelled $u^i(k)$. The choice of this objective function results in the robots not stalling and moving to their goal position as fast as possible (minimum time solution).

5.2.3 Path (Kinematic) Constraints

Constraints (5.2)-(5.6) define the path of each robot. The constraints (5.2), (5.3), and (5.4) form the boundary conditions. Constraint (5.2) indicates that each robot i has to start at a designated start point o^i. Constraint (5.3) initializes the arc length travelled u to zero value. Constraint (5.4) provides the upper bound on the arc length travelled. Constraint (5.5) increments the arc length at each time step based on the speed of the robot ($\Delta t = 1$). Constraint (5.6) ensures that the robots follow their respective paths as defined by the cubic splines. The function $ps^i(u^i(k))$ denotes the location of robot i at time step k, $\forall k \in \{t, \dots, t+T_{hor}\}$ after traveling an arc length of $u^i(k)$ along the piecewise cubic spline curves. It should be noted that the constraint (5.6) is a non-convex nonlinear equality constraint.

5.2.4 Speed and Acceleration (Dynamic) Constraint

Constraints (5.7)-(5.8) are dynamic constraints and ensure that the speed $s^i(k)$ (and hence, angular velocity) and the acceleration $a^i(k)$ for each robot $i = 1 \dots n$

at each time-step k, $\forall k \in \{t, \ldots, t + T_{hor}\}$ are bounded from above (by s_{max} and a_{max} respectively) and below (by s_{min} and a_{min} respectively). These constraints are determined by the capabilities of the robot and the curvature of the paths represented by the spline curve. Here we assume that the curvature of the paths is within the achievable bounds of the angular speed and radial acceleration of the robots. Hence the angular speed required by the robots corresponding to the optimal speed is always achievable, and can be determined by (4.8).

5.2.5 Collision Avoidance Constraint

The non-convex constraint (5.10) ensures that there is a sufficiently large distance d_{safe} between each pair of robots to avoid a collision at all times.

5.2.6 Communication Connectivity Constraint

Constraints (5.12) and (5.13) state that vehicle i should be in one-hop communication range of at least n_{conn} vehicles. This means that, for at least n_{conn} values of $j = 1, ..., n$, $j \neq i$, the condition $d^{ij} \leq \eta_d$ should be satisfied. The remaining vehicles may or may not be in one-hop communication range of i. In order to express this requirement, we introduce a constant M and formulate constraint (5.11), which states that if $C^{ij}(k)$= 1 then vehicles i and j are within one-hop communication range. If $C^{ij}(k) = 0$, then the constraint will be trivially satisfied for a sufficiently large M. Constraint (5.11) is an example of a big-M constraint [84].

Note: In our implementation, we used an equivalent form of constraint (16) as $\frac{(d^{ij}(k))^2}{(M(1 - C^{ij}(k)) + \eta_d)} \leq (M(1 - C^{ij}(k)) + \eta_d)$, in order to avoid the nondifferentiability of a Euclidean distance calculation within the nonlinear solver. The nondifferentiability is not going to occur at the optimal solution due to the collision avoidance constraint keeping d^{ij} sufficiently large, but during the initial iterations of the MILANO

solver, it may cause numerical difficulties. The reformulation removes the potential of such an occurrence and provides numerical stability.

5.3 Algorithm

All robots are initially assumed to be in communication range of each other. The general outline of the algorithm is as follows:

For any time step t, let each robot i implement the following algorithm:

Start: Start at time t

- **Step 0** - An order is enforced in terms of which robot plans its trajectories first. The ordering can be randomly assigned or can be assigned a priori.

- **Step 1** - Based on its decision order $ord(i)$, each robot i solves the problem $\mathcal{O}^i(t+1)$ at time t by taking into account the following plans:

  1.1 Plans $\mathcal{P}^j(t+1)$ for robots j, $\forall j \in \{1, 2, \ldots, n\}$, $j \neq i$ whose $ord(j) < ord(i)$ - these robots have already calculated their new plans, and

  1.2 Plans $\mathcal{P}^\zeta(t)$ for robots ζ, $\forall \zeta \in \{1, 2, \ldots, n\}$, $\zeta \neq i$ whose $ord(i) < ord(\zeta)$ - these robots are yet to calculate their new plans.

- **Step 2** -

  2.1 If a feasible solution is found, the new plan is $\mathcal{P}^i(t+1)$.

  2.2 If $\mathcal{O}^i(t+1)$ is infeasible then use the previously available plan $\mathcal{P}^i(t)$ for the next $T_{hor} - 1$ time steps i.e. the new plan

$$\mathcal{P}^i(t+1) = \mathcal{P}^i(t) \backslash \mathbf{p}^i(t) \tag{5.14}$$

  where $\mathbf{p}^i(t) = (x^i(t), y^i(t))$

- **Step 3** - Broadcast this plan to the other robots.

End: End by $t + 1$, and repeat

5.3.1 Remarks about the algorithm

In our numerical testing, we found that the decision order $ord(i)$ of the robots $i = 1, \ldots, n$ can qualitatively affect the solution of each robot depending on the geometry of the paths.

1. Due to the inherent decentralized decision making, certain robots' decisions may render the coordination problem difficult to solve for other robots. In some cases, reassigning a different decision order $ord(i)$ of the robots $i = 1, \ldots, n$ helped improve overall solutions.

2. Also in some cases, certain robots' decisions can render the coordination problem infeasible for other robots regardless of the decision ordering used. In such cases, the robots may use their plans from the previous time steps as indicated by Step 2.2 of the algorithm above.

6. Numerical Results

We performed extensive numerical simulations to primarily understand the effect of the communication constraints on the velocity profile design. Additionally, we investigate the scalability (in the number of robots) of the proposed models.

In this chapter we provide the results of the numerical testing for the MINLP models. We tested scenarios involving up to ten (10) robots to demonstrate (i) the effect of communication connectivity requirements on robot velocity profiles; and (ii) the dependence of the scenario completion time on communication connectivity requirements.

6.1 Simulation Setup

1. The MINLP models were implemented in `MATLAB` using the symbolic toolbox. The solver `MILANO` was used to perform numerical optimization. The `MATLAB-MILANO` combination was implemented on Mac OS X version 10.6.8 running on a 2.4 Ghz Intel Core 2 Duo processor with 4GB of main memory.

2. The NLP model was implemented in the modeling environment `AMPL` and the solver `LOQO` was used. The `AMPL-LOQO` combination was implemented on a PC running RedHat Linux 2.4.20-8 with 512MB of main memory and a 2.4GHz clock speed. We used `LOQO` Version 6.07 compiled with the `AMPL` solver interface Version 20021031.

3. Paths for each of the robots were generated randomly in `MATLAB`, using 6 waypoints for each robot. The function `spline()` was used to generate piecewise cubic splines passing through the waypoints, parametrized by arc length u. Finally the individual splines were combined to generate the spline curve. The

spline curve paths of the robots were plotting with different colors indicating different robots.

4. The origin and destination points of each robot are indicated by a dot marking. The triangular markings on the curves indicate the positions of the robot at each step in time while traveling at optimal speeds along the path.

5. In all plots, the triangular markings on different paths do not overlap with each other completely at any point in time. This observation indicates that the robots indeed do not collide with each other at any point in time (satisfying the collision avoidance constraint at all times).

6.2 Model Implementation Information

For the `MATLAB-MILANO` setup, the decision variable for the centralized MINLP formulation (4.42)-(4.57) has $Tn^2 + 4Tn + 1$ elements (n robots, T discrete time steps). The number of constraints for this problem is given by $4Tn^2 + 12Tn + n$. The number of decision variables and constraints grow polynomially in n.

On the other hand, for the decentralized formulation (5.1)-(5.13), for each robot $i = 1 \dots n$ there are $t_{hor}(n + 3)$ decision variables as shown here. The number of

Table 6.1: Problem size comparison for centralized and decentralized MINLP formulations for n robots and t_{hor} time steps

	No. of decision variables	No. of binary variables	No. of constraints
Centralized	$t_{hor}n^2+4t_{hor}n+1$	n^2t_{hor}	$4t_{hor}n^2 + 12t_{hor}n + n$
Centralized (no T_{max})	$t_{hor}n^2 + 3t_{hor}n$	$(n^2 - n)t_{hor}$	$4t_{hor}n^2 + 9t_{hor}n$
Decentralized	$t_{hor}n + 3t_{hor}$	$(n - 1)t_{hor}$	$4t_{hor}n + 9t_{hor}$

constraints for each robot $i = 1 \dots n$ for the decentralized scenario is $4t_{hor}n + 9t_{hor}$. As can be seen, the number of decision variables and constraints for each robot i grows linearly in the number of robots n.

Table 6.1 lists the number for decision variables for a n robot scenario for t_{hor} time steps for both centralized and decentralized cases.

The decision variable for the centralized formulation is given by

$$
\begin{bmatrix}
x^{i=1\dots n}(1\dots T)\\
y^{i=1\dots n}(1\dots T)\\
u^{i=1\dots n}(1\dots T)\\
s^{i=1\dots n}(1\dots T)\\
A^{i=1\dots n}(1\dots T)\\
C^{ij,i\neq j}(1\dots T)
\end{bmatrix}
=
\begin{bmatrix}
\left.\begin{matrix} x^{i=1\dots n}(1)\\ \vdots \\ x^{i=1\dots n}(T)\end{matrix}\right\}[n\times T]\\
\\
\left.\begin{matrix} y^{i=1\dots n}(1)\\ \vdots \\ y^{i=1\dots n}(T)\end{matrix}\right\}[n\times T]\\
\\
\left.\begin{matrix} u^{i=1\dots n}(1)\\ \vdots \\ u^{i=1\dots n}(T)\end{matrix}\right\}[n\times T]\\
\\
\left.\begin{matrix} s^{i=1\dots n}(1)\\ \vdots \\ s^{i=1\dots n}(T)\end{matrix}\right\}[n\times T]\\
\\
\left.\begin{matrix} A^{i=1\dots n}(1)\\ \vdots \\ A^{i=1\dots n}(T)\end{matrix}\right\}[n\times T]\\
\\
\left.\begin{matrix}
\left.\begin{matrix} C^{i1}(1)\\ \vdots \\ C^{i1}(T)\end{matrix}\right\}[n\times T]\\
\left.\begin{matrix} C^{i(n-1)}(1)\\ \vdots \\ C^{i(n-1)}(T)\end{matrix}\right\}[n\times T]
\end{matrix}\right\}[(n-1)\times n\times T]\\
\\
\left.\begin{matrix} T_{\max}\end{matrix}\right\}1
\end{bmatrix}_{[n\times(n+4)\times T+1]}
$$

The decision variable for the decentralized formulation is given as

$$
\begin{bmatrix} x^i(t \dots t_{hor}) \\ y^i(t \dots t_{hor}) \\ u^i(t \dots t_{hor}) \\ s^i(t \dots t_{hor}) \\ C^{ij}(t \dots t_{hor}) \end{bmatrix} =
\begin{bmatrix}
\left.\begin{matrix} x^i(t) \\ \vdots \\ x^i(t+t_{hor}) \end{matrix}\right\} t_{hor} \times 1 \\
\left.\begin{matrix} y^i(t) \\ \vdots \\ y^i(t+t_{hor}) \end{matrix}\right\} t_{hor} \times 1 \\
\left.\begin{matrix} u^i(t) \\ \vdots \\ u^i(t+t_{hor}) \end{matrix}\right\} t_{hor} \times 1 \\
\left.\begin{matrix} s^i(t) \\ \vdots \\ s^i(t+t_{hor}) \end{matrix}\right\} t_{hor} \times 1 \\
\left.\begin{matrix}
\left.\begin{matrix} C^{i1}(t) \\ \vdots \\ C^{i1}(t+t_{hor}) \end{matrix}\right\} t_{hor} \times 1 \\
\vdots \\
\left.\begin{matrix} C^{i(n-1)}(t) \\ \vdots \\ C^{i(n-1)}(t+t_{hor}) \end{matrix}\right\} t_{hor} \times 1
\end{matrix}\right\} [(n-1)\mathrm{x}(t_{hor})]\mathrm{x}1
\end{bmatrix}_{[(n+3)\times(t_{hor})]\times 1}
$$

Table 6.2: Centralized MINLP model parameter values used for simulations

d_{safe}	0.02 m	s_{min}	0	s_{max}	2 m/s
a_{min}	-1 m/s^2	a_{max}	0.5 m/s^2	$M1, M2$	10

6.3 Scenarios Tested

We tested our model for scenarios that included up to ten (10) mobile robots and a number of communications constraints. The parameters used in our simulations are listed in Table. 6.2. For all the simulations the value of $\Delta t = 1$.

6.3.1 Centralized vs. Decentralized Formulations

In this section, we compare the solution computation times for a 2 robot scenario in both centralized (MINLP) and decentralized (MINLP) settings. We test the scenarios for $n_{conn} = 0$ and $n_{conn} = 1$.

- *Centralized Scenario*: For a scenario where $\eta_d = 2.5$ m, Figure 6.1 shows the trajectories and velocity profiles of the 2 robots for $n_{conn} = 0$ (no communication connectivity requirement) and $n_{conn} = 1$. No changes are observed in the trajectories and velocity profiles of both robots as n_{conn} goes from 0 to 1. T_{max} remains 7 for both $n_{conn} = 0$ and $n_{conn} = 1$.

- Decentralized Scenario: For a scenario where $\eta_d = 2.5$ m, Figure 6.2 shows the trajectories of the 2 robots for $n_{conn} = 0$ (no communication connectivity requirement) and $n_{conn} = 1$. It is observed, that the trajectories of both robots change as n_{conn} goes from 0 to 1. Figure 6.3 shows the velocity profiles of both

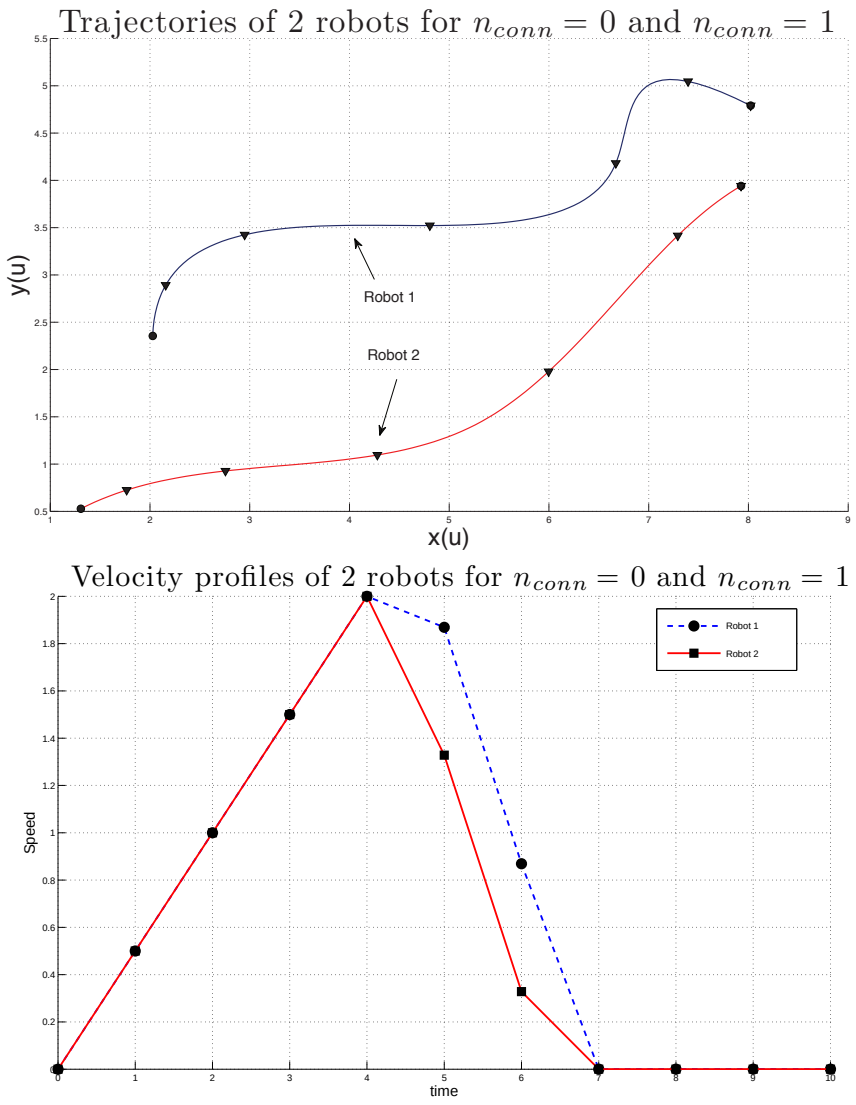

Figure 6.1: Top figure indicates the trajectories of both robots for $n_{conn} = 0$ and $n_{conn} = 1$. Bottom figure indicates the velocity profiles of both robots

robots for these two scenarios. T_{max} increases from 7 to 10 as n_{conn} goes from 0 to 1.

As can be seen for n_{conn} =1, T_{max} increases from 7 for centralized planning scenario to 10 for the decentralized planning scenario. Such trade-offs are not uncommon while switching from centralized to decentralized planning.

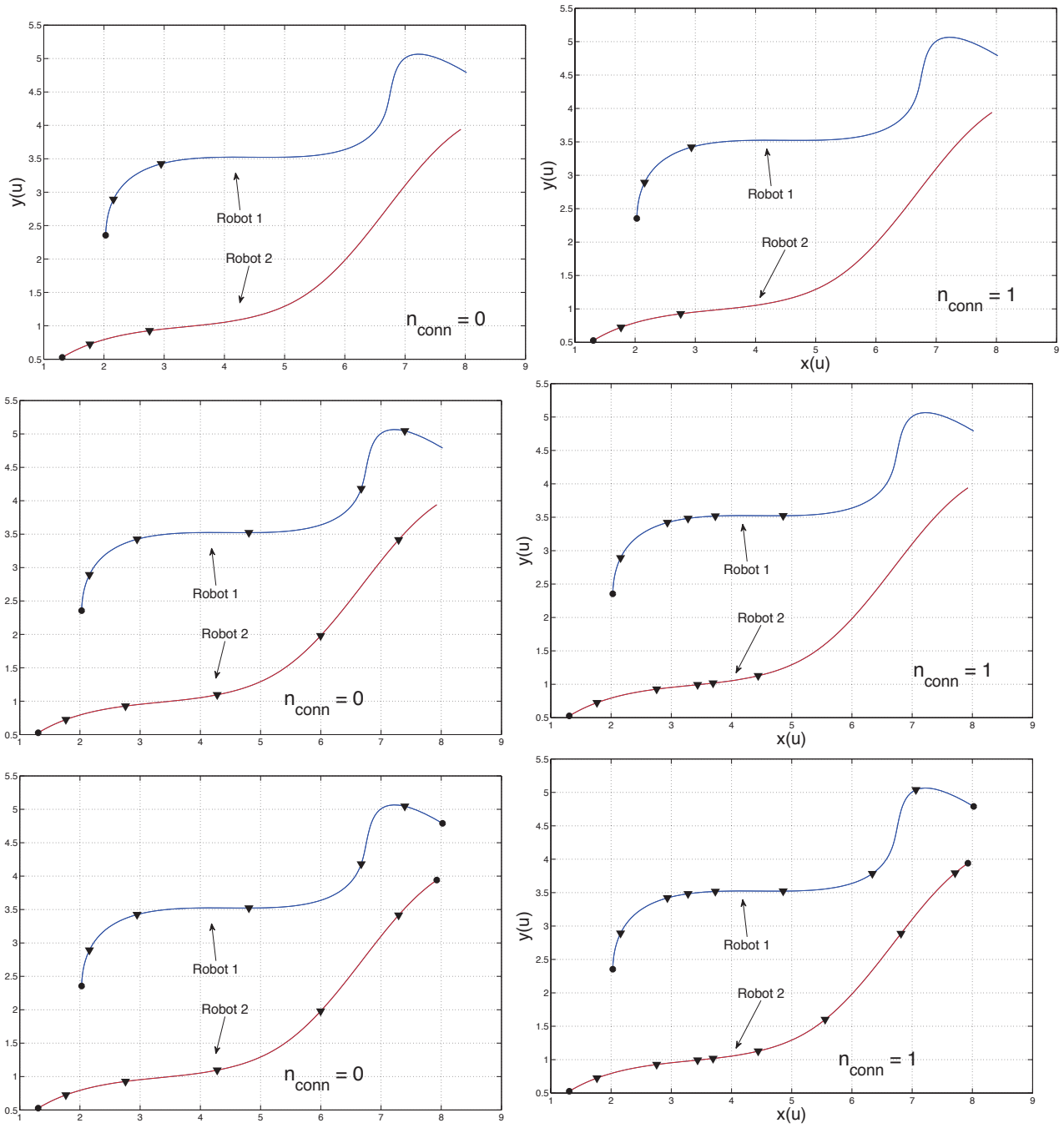

Figure 6.2: The figures in the left column indicate robot trajectories for $n_{conn} = 0$ - both robots reach their destination by $T_{max} = 7$. The figures in the right column indicate robot trajectories for $n_{conn} = 1$. $T_{max} = 10$ for this case.

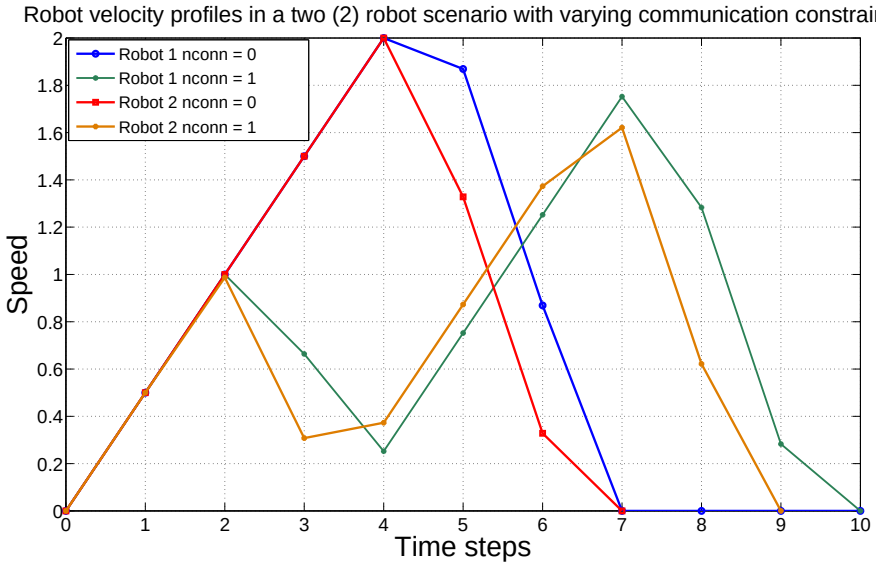

Figure 6.3: Velocity profiles of robots in a 2 robot scenario with varying communication constraints

Table 6.3: Decentralized MINLP model parameter values used for simulations

d_{safe}	0.02 m	s_{min}	0	s_{max}	2 m/s
a_{min}	-1 m/s^2	a_{max}	0.5 m/s^2	M	10

6.4 Decentralized Scenarios Tested

For the decentralized formulation as well, we tested our model for scenarios that included up to ten (10) mobile robots and a number of communications constraints. The parameters used in our simulations are listed in Table. 6.3. For all the simulations the value of $\Delta t = 1$ and $T_{hor} = 5$.

6.4.1 Decentralized Formulation: Effect of varying n_{conn} on the velocity profiles

We vary n_{conn} between 0 to $n-1$ for $n = 6$ and 10 and demonstrate the effect on velocity profiles.

- *6 robots*: For a scenario where $\eta_d = 3$ m, Figure 6.4 shows the trajectories of the 6 robots for $n_{conn} = 1$ and $n_{conn} = 3$. It is observed, that the trajectories of Robot 1 and Robot 2 change as n_{conn} goes from 1 to 3. T_{max} remains unchanged at 10 as n_{conn} goes from 1 to 3.

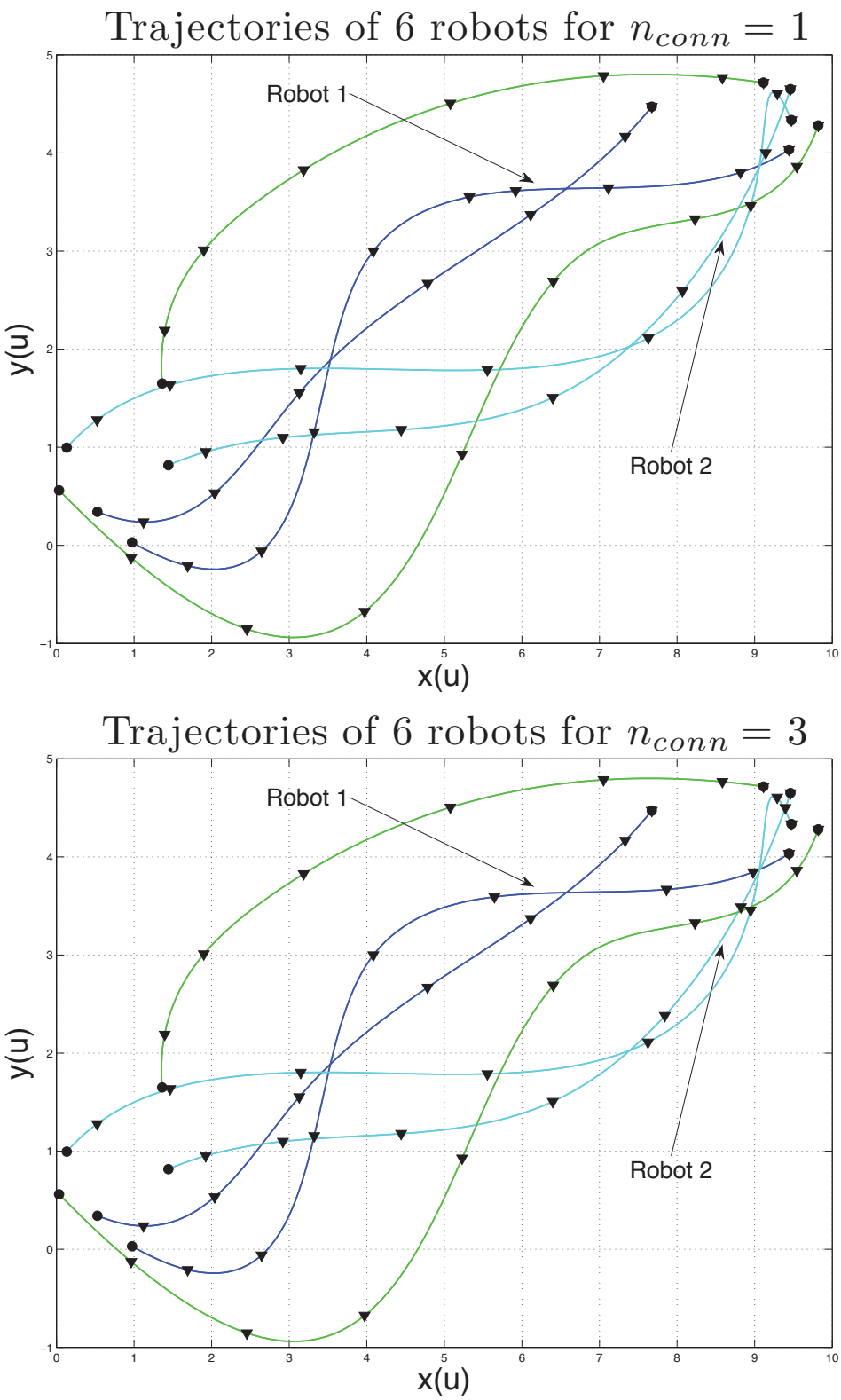

Figure 6.4: A 6 robot scenario with varying communication constraints

- *10 robots*: For a scenario where $\eta_d = 4$ m, Figure 6.5 shows the trajectories of the 10 robots for $n_{conn} = 0$ (no communication connectivity requirement) and $n_{conn} = 9$. The most visible trajectory changes are observed in Robot 1 and Robot 2 trajectories. T_{max} remains unchanged at 10 as n_{conn} goes from 0 to 9.

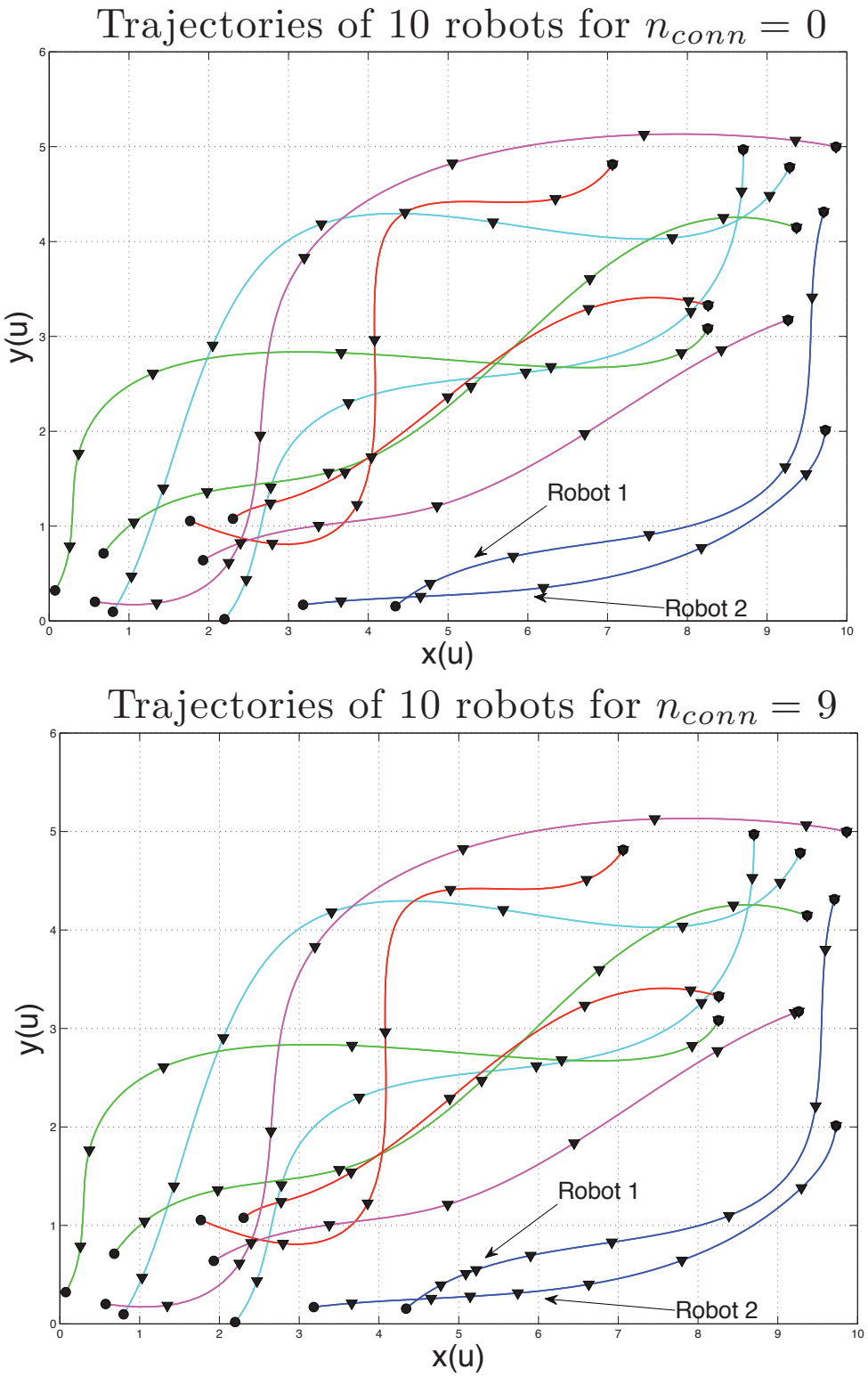

Figure 6.5: A 10 robot scenario with varying communication constraints

Table 6.4: Decentralized MINLP Formulation: T_{comp} in seconds for various scenarios

	n_{conn}			
n	0	1	3	9
2 ($\eta_d = 2.5$ m)	21.3	23.2	-	-
6 ($\eta_d = 3$ m)	308.9	348.1	355.8	-
10 ($\eta_d = 5$ m)	983	937.2	2142.5	1093.5

6.4.2 Decentralized Formulation: Effect of varying n_{conn} on solution computational time:

For calculating the solution computation time T_{comp}, we measure the total time taken by all robots $i = 1 \dots n$ to solve $\mathcal{O}^i$ for all discrete time steps till the scenario is completed. All other MATLAB-MILANO processing times are ignored. The results are listed in Table 6.4. It is observed that for the scenarios where $n = 2$ and $n = 6$, T_{comp} steadily increases with an increase in n_{conn}. However, in the case when $n = 10$, T_{comp} varies unpredictably. As, mentioned earlier in subsection 5.3.1, this is due to the fact that certain robots' decisions may render the coordination problem difficult for other robots. In such cases, the value of T_{comp} can increase drastically as observed in the case when $n = 10$ and $n_{conn} = 3$ ($T_{comp} = 2142.5$ seconds).

MILANO issues infeasibility certificates if a robot's problem has no feasible solution. When such a situation is encountered, reassigning a different decision order $ord(i)$ of the robots $i = 1, \dots, n$ can result in improved T_{comp} values. In the case where $n = 10$ and $n_{conn} = 9$, a decision order different than the case where $n = 10$ and $n_{conn} = 3$ was used. It is observed that the resulting value of T_{comp} improves considerably ($T_{comp} = 1093.5$ seconds).

7. Conclusions and Future Directions

In this chapter, we summarize the details of the technical developments documented in this thesis. We also attempt to codify the implications of this work on future research.

7.1 Summary of work

The work in this thesis commenced with the goal of using MP techniques to model and solve MVMP problems. It was our belief that MP techniques are general enough to capture most if not all constraints arising in MVMP problems. In the following sections, we summarize the conclusions that were drawn from our investigations.

7.1.1 Communication Constraint Modeling

In our communication constraint formulations, we attempted to capture the details of the fundamental physical phenomenon that facilitate successful wireless transmissions. Towards this end we used well-established deterministic and stochastic models that specified the received power as a function of the transmitted power, the Euclidean distance between the transmitter and receiver, and the random Shadowing effects experienced due to environmental factors. Using indicator variables and Big-M method, we were able to encapsulate the discrete on or off states of the wireless communication and enforce communication connectivity requirements. Furthermore, we developed *Partition Elimination Constraints* that ensure that when feasible, the communication network remained unpartitioned.

In conclusion, we developed novel MP formulations that captured physical layer characteristics of wireless communication and enforced communication connectivity

requirements.

7.1.2 Models for Multi-Vehicle Path Coordination under Communication Constraints

The variant of MVMP that we investigated using MP techniques was Multi-Vehicle Path Coordination. In this problem, we generated speed profiles for multiple path constrained mobile robotic vehicles to minimize transit times. The vehicles had bounds on their kinematics and dynamics, and were required to avoid collisions. Most importantly, these vehicles were required to satisfy communication connectivity requirements.

We modeled this scenario using both continuous and discrete variables to capture the fundamental physical principles involved in motion and wireless communication. Additional constraints on collision avoidance and communication connectivity were formulated and introduced in the problem. Mindful of the fact that the number of *Partition Elimination Constraints* increase exponentially in the number of robots, we devised the P.E. algorithm to only introduce these constraints when network partitions were detected. This approach greatly reduced the number of constraints included in the problem at hand resulting in a reduced computational effort. For the decentralized case, we presented a RH-MINLP formulation that was embedded in a sequential, decentralized decision making algorithm.

In conclusion, we developed novel MP models that capture the details of the problem of Multi-Vehicle Path Coordination under Communication Constraints.

7.1.3 Formalization of Feasibility Conditions

It is important to understand the conditions that can lead infeasibility or unboundedness in a MP problem. We attempted to understand conditions that would

lead to infeasibility in the problem of Multi-Vehicle Path Coordination under Communication Constraints. We established the fact that the communication constraints are the primary source of infeasibility in the problem. Without communication constraints, a solution always exists. We also identified and formalized two conditions where satisfying communication constraints would require violating kinematic and dynamic constraints respectively. For the sequential decentralized algorithm, the fact that the decision order can affect feasibility was verified by our simulations.

In conclusion, we identified communication connectivity constraints as the source of infeasibility in the problem of Multi-Vehicle Path Coordination under Communication Constraints and defined two conditions that lead to infeasibility in the problem.

7.1.4 Numerical Results

We generated numerical results for scenarios involving up to 50 robots using `MATLAB`, `AMPL`, `LOQO`, and `MILANO`. We tested the models with several communication connectivity constraints, and enlisted the problem size and solution computation times.

In conclusion, the communication constraints affected the speed profiles of the vehicles and in many cases the vehicles slowed down to accommodate more stringent communication connectivity requirements. In our numerical testing, `AMPL` *performed symbolic computations much faster than* `MATLAB`*.*

7.2 Future Directions

In this section, we list some of the promising and exciting future direction.

1. The communication models can be extended to capture more details of the wireless network state. Several details of the communication protocol and network topology, if modeled correctly and efficiently, can assist in effectively specifying

high level Quality of Service (QoS) requirements. An obvious extension is to use graph theoretic results. Recent work by *Derenick et. al* [56] in this direction is promising.

2. Further investigation of feasibility conditions for MVMP problems is necessary. Specifically, finding conditions that will lead to provable feasibility of the problem of Multi-Vehicle Path Coordination under Communication Constraints under all possible scenarios will be immensely useful for the community. The conditions developed in this thesis along with safety proofs developed by *Schouwenaars et al.* in [13] may serve as useful guides towards this end.

3. Recent work by *Mattingley and Boyd* [116] and *Wang and Boyd* [117] among others is evidence of the fact that MP solution algorithms can be applied in real time systems. The ultimate test of the proposed framework will be when it is applied to real world systems. This will require a collaborative effort between researchers and engineers belonging to both the MP and MVMP communities.

Appendices

Appendix A. NLP reformulation

The robust MINLP formulation (4.42)-(4.58) was reformulated to eliminate integer variables. The resulting nonlinear programming formulation is given by (A.1) - (A.17).

A.1 Model Formulation

The following features of the (A.1) - (A.17) model are to be noted:

1. The kinematic constraints, dynamic constraints, and collision avoidance constraints do not involve any integer variables and hence remain unchanged.

2. An equivalent formulation of $T_{\max}$ has been developed to eliminate the need of using integer variables.

3. The communication constraint has been reformulated to eliminate the use of integer variables.

$$\text{minimize} \quad T_{\max} + \sigma \sum_{i,t} d^i_{goal}(t) \tag{A.1}$$

$$\forall i \in \{1,2,...n\}, \forall t \in \{1,2,...T\}, \forall j \in \{1,2,...n\}, j \neq i$$

$$\text{subject to} \quad (x^i(0), y^i(0)) = o^i \tag{A.2}$$

$$(x^i(T), y^i(T)) = e^i \tag{A.3}$$

$$u^i(0) = 0 \tag{A.4}$$

$$u^i(t) = u^i(t-1) + s^i(t)\Delta t \tag{A.5}$$

$$(x^i(t), y^i(t)) = ps^i(u^i(t)) \tag{A.6}$$

$$s_{min} \leq s^i(t) \leq s_{max} \tag{A.7}$$

$$a_{min} \leq a^i(t) \leq a_{max} \tag{A.8}$$

$$d^{ij}(t) \geq d_{safe} \tag{A.9}$$

$$d^i_{goal}(t) = U^i - u^i(t) \tag{A.10}$$

$$0 \leq A^i(t) \leq 1 \tag{A.11}$$

$$A^i(t) d^i_{goal}(t) = 0 \tag{A.12}$$

$$\forall i, T_{\max} \geq \left(\sum_{t=0}^{T} \left(1 - A^i(t)\right) \right) \tag{A.13}$$

$$0 \leq C^{ij}(t) \leq 1 \tag{A.14}$$

$$l^{ij}(t) = \text{SNR}^{ij}_r(t) - \eta_{snr} \tag{A.15}$$

$$C^{ij}(t) l^{ij}(t) \geq 0 \tag{A.16}$$

$$\sum_{j:j \neq i} C^{ij}(t) \geq n_{conn} \tag{A.17}$$

A.1.1 Definition of $T_{\max}$

As defined by constraints (A.11) and (A.12), $A^i(t)$ measures the number of time periods for which the robot is not at the destination. The equilibrium constraint (A.12) and the bounds on $A^i(t)$ (A.11) ensure that when $d^i_{goal}(t) > 0$, the value of $A^i(t) = 0$. Therefore, if $A^i(t) = 1$ for all (i, t) with $d^i_{goal}(t) = 0$, the total amount of time it takes a robot i to reach its destination can be calculated using (A.18).

$$\sum_{t=0}^{T} \big(1 - A^i(t)\big) \tag{A.18}$$

The equilibrium constraint (A.12) cannot alone guarantee that this property will hold. However, constraint (A.13) specifies $T_{\max}$ as an upper bound for this sum, and equation (A.1) minimizes $T_{\max}$.

Therefore at the optimal solution, for the last robot(s) to reach its destination, $A^i(t) = 1$ when robot i is at its destination at time t and that (A.13) will hold with equality.

Note that the solution obtained by including the constraints (A.11) - (A.13) is equivalent to the one obtained by using the mixed-integer definition (4.23) - (4.24):

$$\begin{aligned} A^i(t) &= 0 \text{ if } d^i_{goal}(t) \neq 0 \\ &= 1 \text{ if } d^i_{goal}(t) = 0 \\ T_{\max} &= \max_{i=1,\ldots,n} \left(\sum_{t=0}^{T} \big(1 - A^i(t)\big) \right) \end{aligned}$$

It is, however, more advantageous for efficiency of the solution algorithm to solve an NLP instead of a MINLP. With recent research in handling equilibrium constraints in NLPs, handling the resulting non-smoothness is not a complicating factor in the

solution process. For details on how the solver handles equilibrium constraints, see [126].

A.1.2 Communication Constraint

Constraints (A.14)-(A.17) define the requirement that each robot must be in communication with at least n_{conn} other robots at all times. Constraint (A.15) defines an intermediate variable, $l^{ij}(t)$, that aids in defining the communications constraint. The sign of $l^{ij}(t)$ at any given point in time indicates whether the robots i and j are within one-hop communication range of one other: $l^{ij}(t) \geq 0$ indicates that the two robots are in communication range whereas $l^{ij}(t) < 0$ indicates that the two robots are not in one-hop communication range of each other.

Constraint (A.16) then ensures that if robots i and j are not in one-hop communication range of each other at time t, then the variable $C^{ij}(t)$ must necessarily equal 0. That is, if pairwise communication is lost, we have that $l^{ij}(t) < 0$ and since constraint (A.14) requires that the value of $C^{ij}(t) \geq 0$, the only way to satisfy constraint (A.16) is to have $C^{ij}(t) = 0$. If the two robots are in one-hop communication range of each other at time t, then $C^{ij}(t)$ can take on any value between 0 and 1, inclusive, as allowed by constraint (A.14).

Finally, constraint (A.17) ensures that for each robot i at time t, at least n_{conn} of the $C^{ij}(t)$, $j \in \{1, 2, ...n\}$, $j \neq i$ must be greater than zero.

There are several issues to consider here:

- This formulation avoids the use of a binary variable to define the communications constraint. Each variable $C^{ij}(t)$ is continuous and bounded below by 0 and above by 1. Doing so greatly reduces the complexity of the problem.

- For any pair of robots that are in communication at time t, there may be an infinite number of optimal values for $C^{ij}(t)$. As an example, assume that in

the optimal solution, robot i is within communication range of robots j and m. Then, as long as $C^{ij}(t) + C^{im}(t) \geq 1$, the communication constraint will be satisfied. Optimal values for $(C^{ij}(t), C^{im}(t))$ include $(0, 1)$, $(1, 1)$, $(1, 0)$, $(0.75, 0.75)$, among others. This does not constitute a difficulty for the solver, since the set of optimal solutions is bounded by constraint (A.14). The values of the variables can be reset to binary values after the optimal solution is found simply by observing the sign of $l^{ij}(t)$.

- Any positive value between 0 and n_{conn} is appropriate for the right-hand side of constraint (A.17) to ensure that each robot is in communication with at least one other robot at all times. However, the value of n_{conn} is chosen to indicate the minimum number of robots with which to require communication. In fact, if this number is changed, for example communication with at least 3 other robots are required at all times, then the right-hand side can simply be changed to 3. Again, this will allow for fractional values of $C^{ij}(t)$, but given that each $C^{ij}(t) \leq 1$ by constraint (A.14), the constraint (A.17) with a right-hand side of 3 cannot be satisfied by establishing communication with less than 3 robots.

- The intermediate variables $l^{ij}(t)$ are provided here to simplify the exposition, but are not necessary to express the same requirements. In fact, constraints (A.15) and (A.16) can be replaced by a single set of constraints of the form (A.19)

$$C^{ij}(t)(\mathrm{SNR}_r^{ij}(t) - \eta_{snr}) \geq 0, j \neq i, j = 1, \ldots, n. \tag{A.19}$$

A.1.3 Jammers

As a special case, we introduce jammer robots in the scenario that attempt to disrupt communication of the non-jamming robots. These jammers can be thought of as stationary or mobile obstacles with an effective jamming radius. Once a robot

(other than a jammer robot) is within the jamming radius, it loses communication with all other robots. Thus the robots must try to stay out of the jamming ranges at all times. We assume that the path and velocity profile of the jammers are known. In reality these will have to be estimated.

Let M be the set of n_{jam} communication jammers, where each jammer $m = 1, 2, \ldots, n_{jam}$ is a mobile robot represented by (4.1) − (4.3) with associated non-holonomic constraint (4.4) and has a fixed path with given start and end points. The effective jamming range of a jammer is d_{jam}. In this study, all the jammers have same jamming radius.

A.1.4 Communication Jamming Constraint

If robot i is within jamming range d_{jam} of one or more jammer robots, it loses all its capabilities to communicate with the other robots. Accordingly constraint (A.20) is added to (A.1)-(A.17) ensure that each robot remains outside the jamming range of the jammer robots at all times.

$$d^{im}(t) \geq d_{jam} \tag{A.20}$$

A.2 Simulation Setup

The optimization model (A.1) - (A.17) along with the P.E. constraints (4.58) was implemented in the modeling environment AMPL and the solver LOQO was used. The AMPL-LOQO combination was implemented on a PC running RedHat Linux 2.4.20-8 with 512MB of main memory and a 2.4GHz clock speed. We used LOQO Version 6.07 compiled with the AMPL solver interface Version 20021031.

Table A.1: NLP model parameter values used for simulations

d_{safe}	0.01 m	s_{min}	0	s_{max}	2 m/s
a_{min}	-1 m/s^2	a_{max}	0.5 m/s^2	σ	100
d_{jam}	0.45 m	Δt	1 s	a_{jam}	0 m/s^2
η_{snr}	4.5 x 10^{-3}	T	10	α	2

A.2.1 NLP Numerical Simulations

For the NLP model, we tested scenarios involving up to fifty (50) robots to demonstrate (i) the trade off between the arrival time and the communication connectivity requirements in scenarios with and without communication jamming; (ii) the effect of communication connectivity requirements on robot velocity profiles; and (ii) the dependence of computation time on the number of robots.

The parameter values used for (A.1) - (A.17) model simulations are listed in Table A.1.

A. **NLP: Effect of the one-hop communication constraint on the velocity profiles**

We vary n_{conn} between 0 to $n-1$ for $n = 2, 6$, and 10 and report on the effects on the velocity profiles.

- *2 robots*:

  For a scenario where $P_{tr} = 2.5$, the top two plots in Figure A.1 show the trajectories of the 2 robots for $n_{conn} = 0$ (no communication connectivity requirement) and $n_{conn} = 1$. It is observed, that the trajectory of the Robot 1 changes as n_{conn} goes from 0 to 1. The bottom plot in Figure A.1 shows the velocity profiles of both the robots for scenarios when $n_{conn} = 0$ and $n_{conn} = 1$. For both the cases, $T_{\max} = 7$.

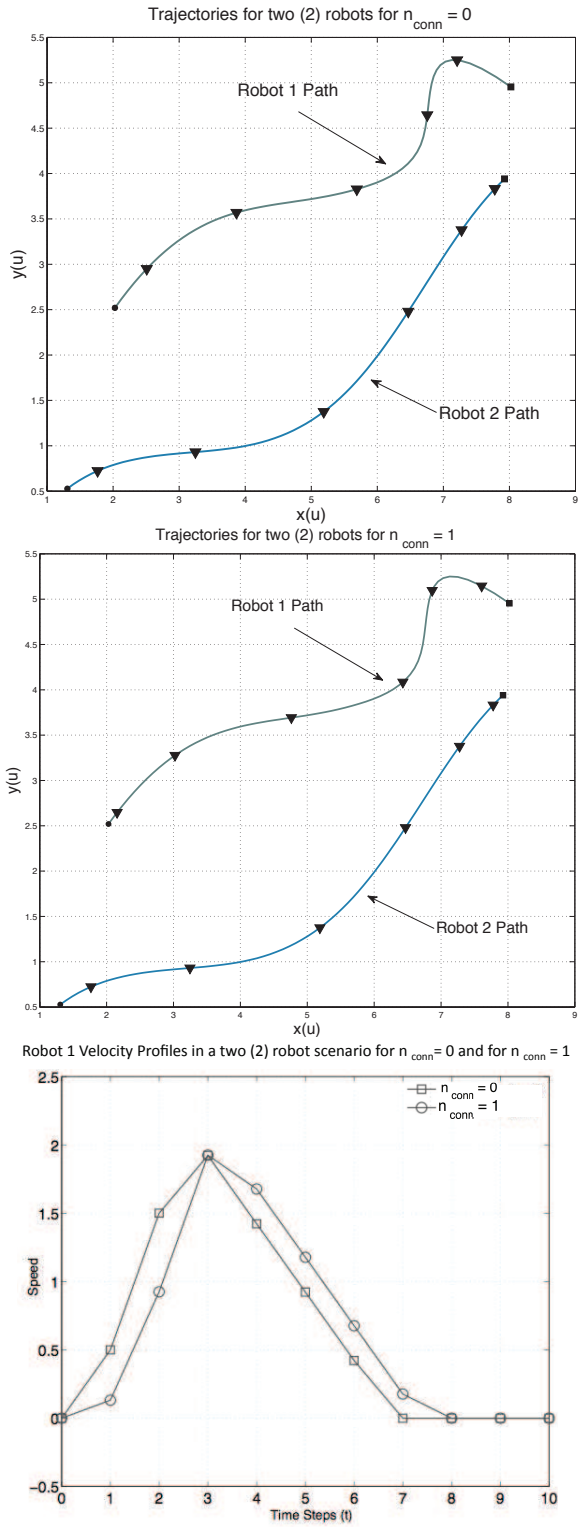

Figure A.1: NLP: A 2 robot scenario with varying communication constraint. Effect of communication constraint on the velocity profiles in a 2 robot scenario

- *6 robots*:

  Figure A.2 shows the trajectories of the 6 robots for scenarios when $n_{conn} = 1$ ($P_{tr} = 2.7$) and $n_{conn} = 5$ ($P_{tr} = 10.1$). The trajectory of the Robot 1 changes with a change in the value of n_{conn}. Figure A.3 shows the velocity profile of the Robot 1 for $n_{conn} = 1$ and $n_{conn} = 5$. For both the cases, $T_{\max} = 9$.

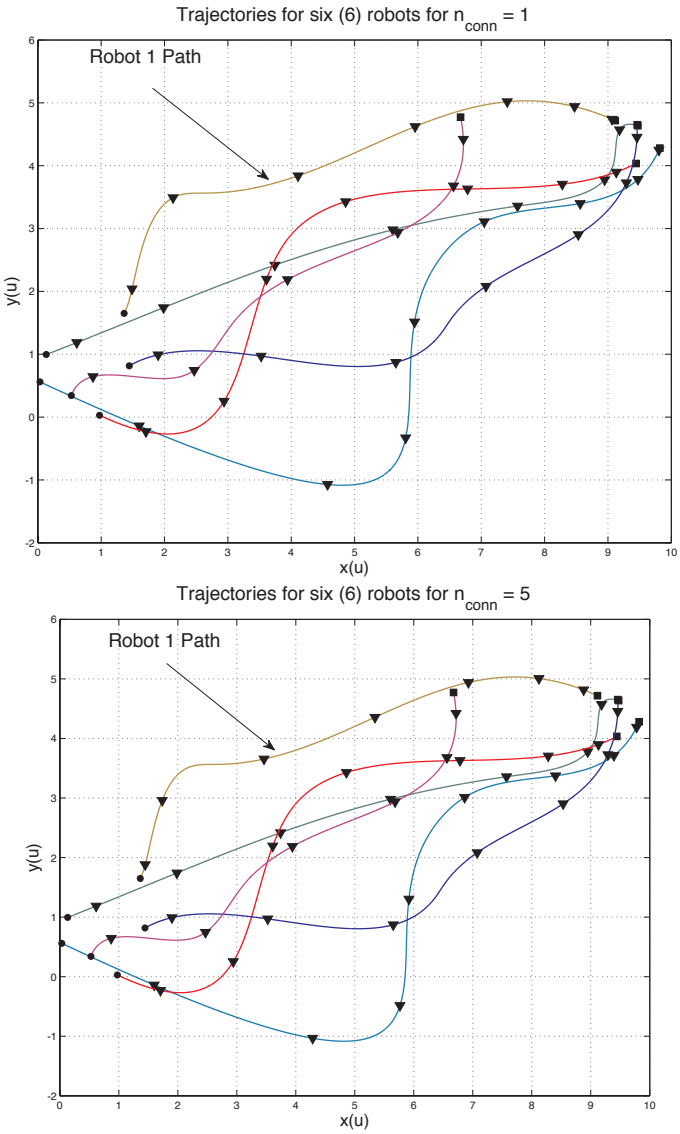

Figure A.2: NLP : A 6 robot scenario with varying communication constraint

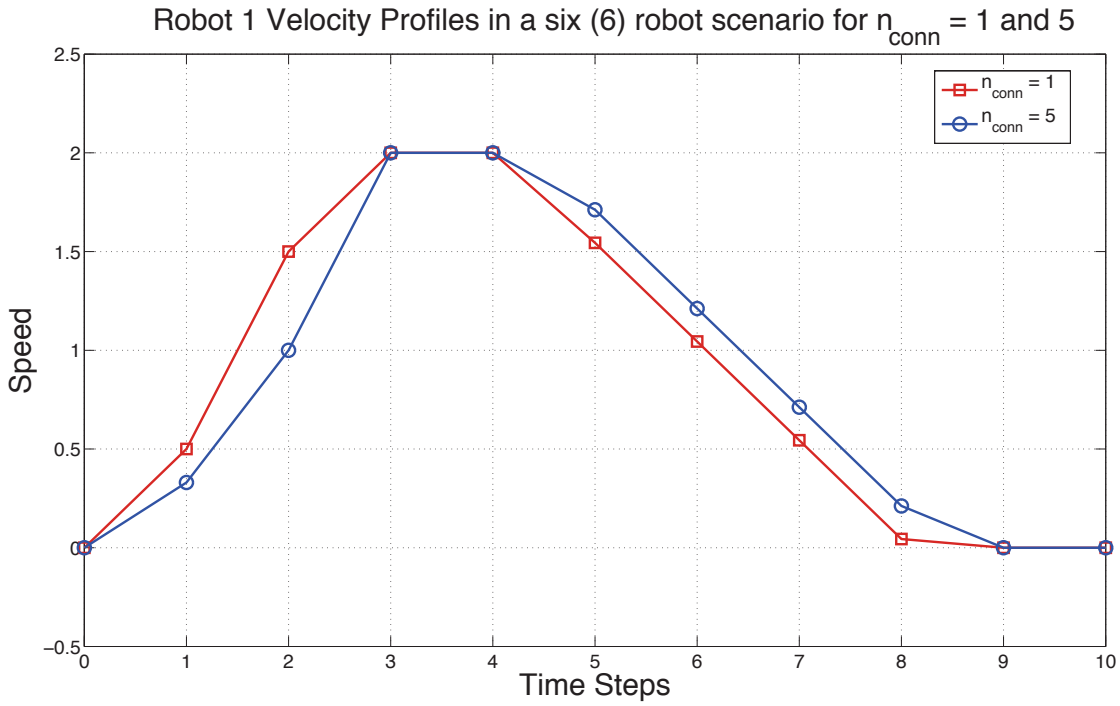

Figure A.3: NLP: Effect of communication constraint on the velocity profiles in a 6 robot scenario

- *10 robots*:

  Figure A.4 shows the trajectories of the 10 robots for scenarios when $n_{conn} = 0$ and $n_{conn} = 9$ ($P_{tr} = 10.5$). The most visible changes in the trajectories that are observed correspond to the Robots 1 and 2. Figure A.5 shows the velocity profile of the Robots 1 and 2 for $n_{conn} = 0$ and $n_{conn} = 9$. For $n_{conn} = 9$, Robot 1 slows down at times steps 2, 3 and 4 as compared to the case when $n_{conn} = 0$ in order to maintain communication with the other robots, but speeds up during the latter part of its journey when its path is closer to the other robots. Similar behavior is observed in case of Robot 2. For both the cases, $T_{\max} = 9$.

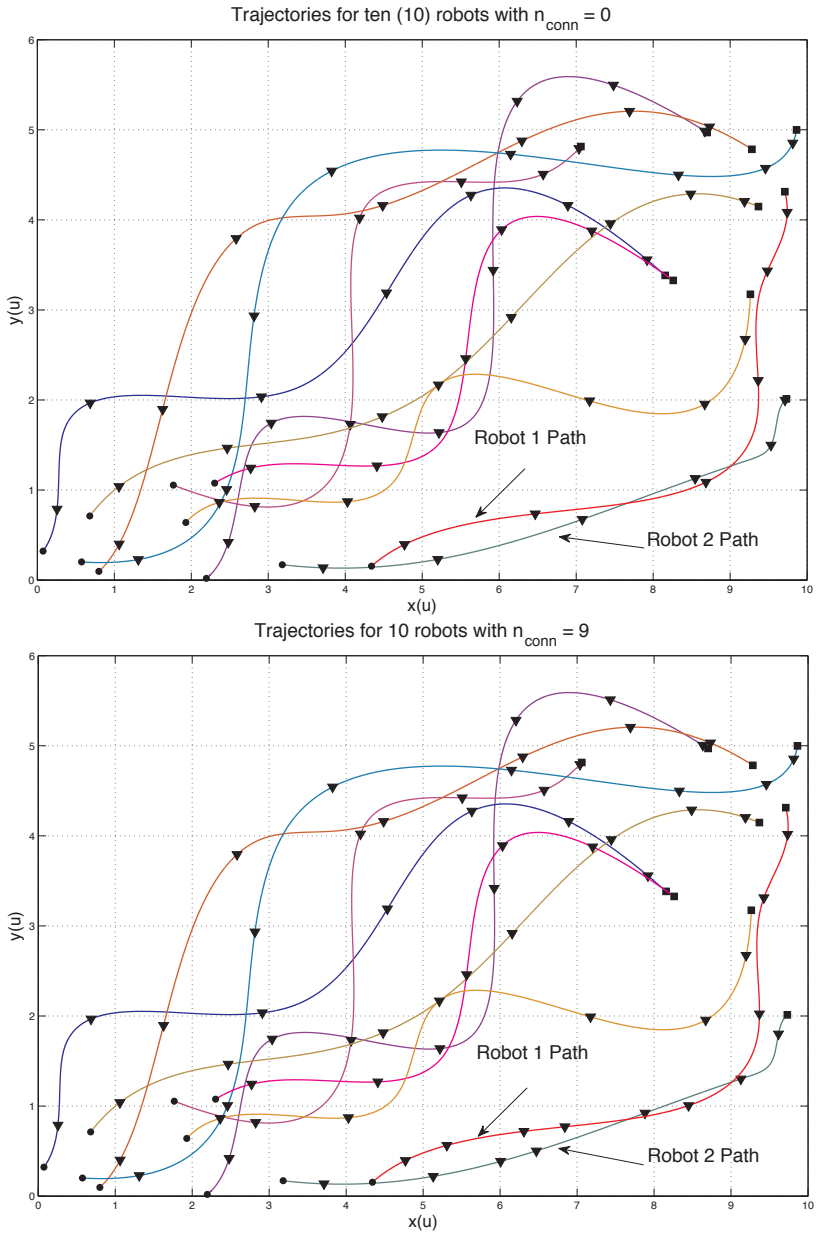

Figure A.4: NLP: A 10 robot scenario with varying communication constraint

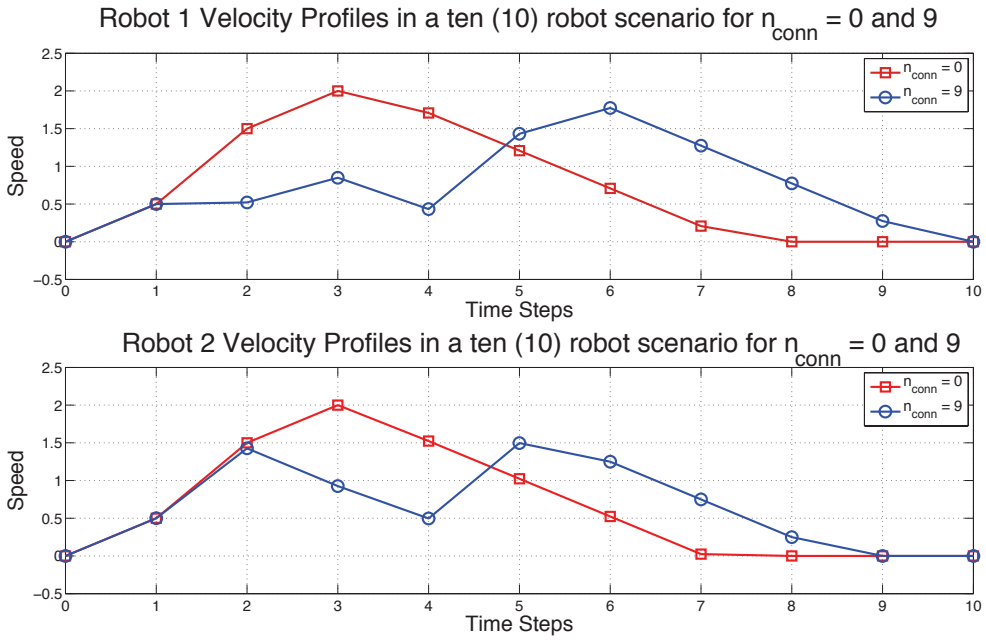

Figure A.5: NLP: Effect of communication constraint on the velocity profiles in a 10 robot scenario

From the above results, the following points are observed

- With an increase in the value of n_{conn}, the velocity profile of the robot(s) change in order to satisfy the communication constraint.
- Even when the communication constraint becomes more stringent, the value of $T_{\max}$ in these examples remained the same, i.e. the cost incurred does not change. Typically, the robots whose times of arrival at their respective destinations are less than $T_{\max}$ change their velocity profiles to comply with the new communication constraint, without affecting $T_{\max}$.

B. **NLP: Effect of the penalty parameter σ on the velocity profiles** We demonstrate the effect of the penalty parameter σ in the objective function for a 2 robot scenario. Figure A.6 indicates the trajectories of the two robots for $n_{conn} = 1$ and $P_{tr} = 2.5$. Clearly for $\sigma = 100$, the robots are much more active as compared to the case where $\sigma = 0$. This results in them ending up closer to the destination at the penultimate time step before reaching the goal as indicated by the circles on both the plots. For both cases $T_{\max} = 7$.

Without penalty, there are infinitely many optimal solutions, each of which satisfies the constraints and has all robots reach their destination within $T_{\max}$. One such solution as depicted on the left side of Figure A.6. With the penalty, however, the robots must travel as close to the destination as possible at each time step, and in this example we show one optimal solution, which is indicated on the right hand side plot of Figure A.6.

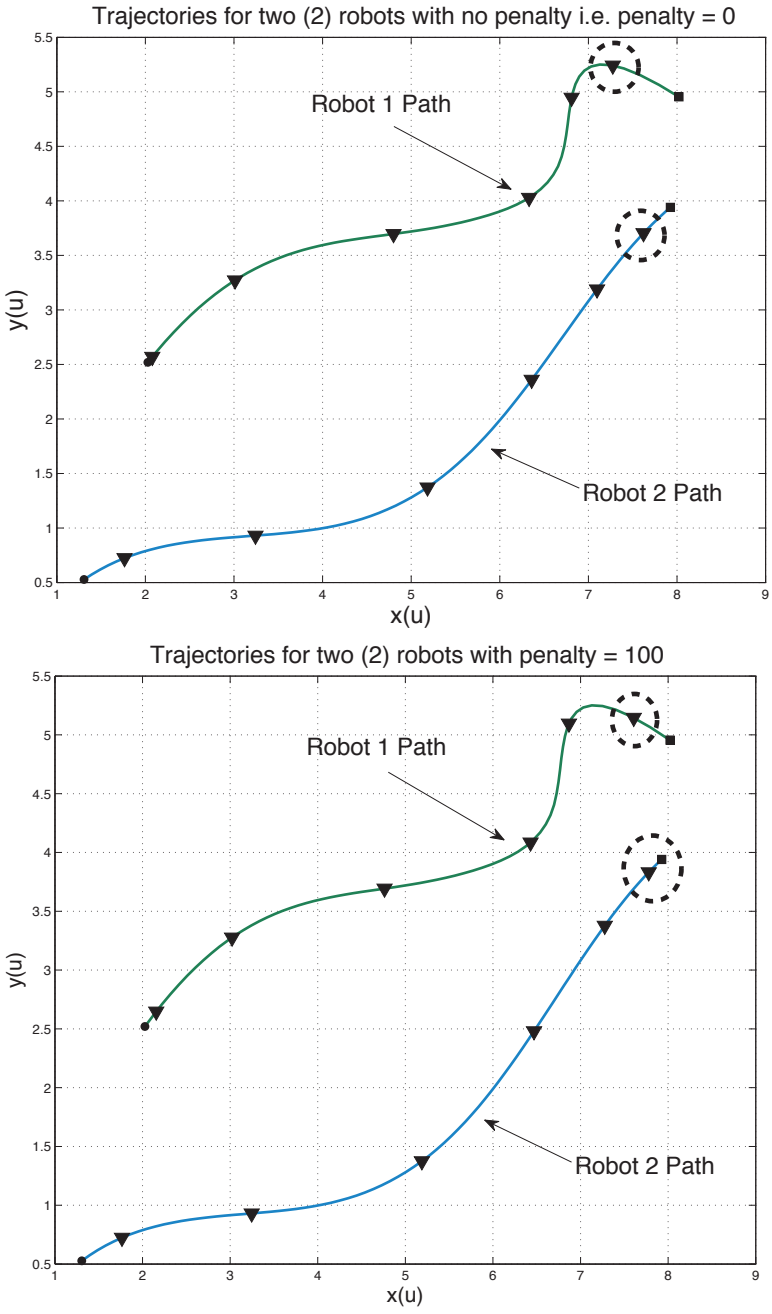

Figure A.6: A 2 robot scenario with and without the penalty term σ for $n_{conn} = 1$

C. **NLP: Effect of partition elimination constraints on the velocity profile**

We used the Partition Elimination (P.E.) algorithm that introduced partition elimination constraints in the NLP -1 C model when network partitions were detected.

- *4 robots*:

  Figure A.7 shows the trajectories of the robots in a four (4) robot scenario in presence (right side plot) and absence (left side plot) of the partition elimination constraints. In absence of the partition elimination constraints, it is observed that all robots travel at the maximum allowed speed at all times. In fact the robots form two subgroups based on the physical proximity of their paths and the one-hop communication constraint is always satisfied. Partitions are detected at time periods 2 and 3.

  After applying the partition elimination algorithm to establish communication between the two subgroups of vehicles at time periods 2 and 3, the trajectory of robot 2 changes. Robot 2 slows down in order to maintain connectivity. In both cases, the mission ends at $T_{\max} = 8$.

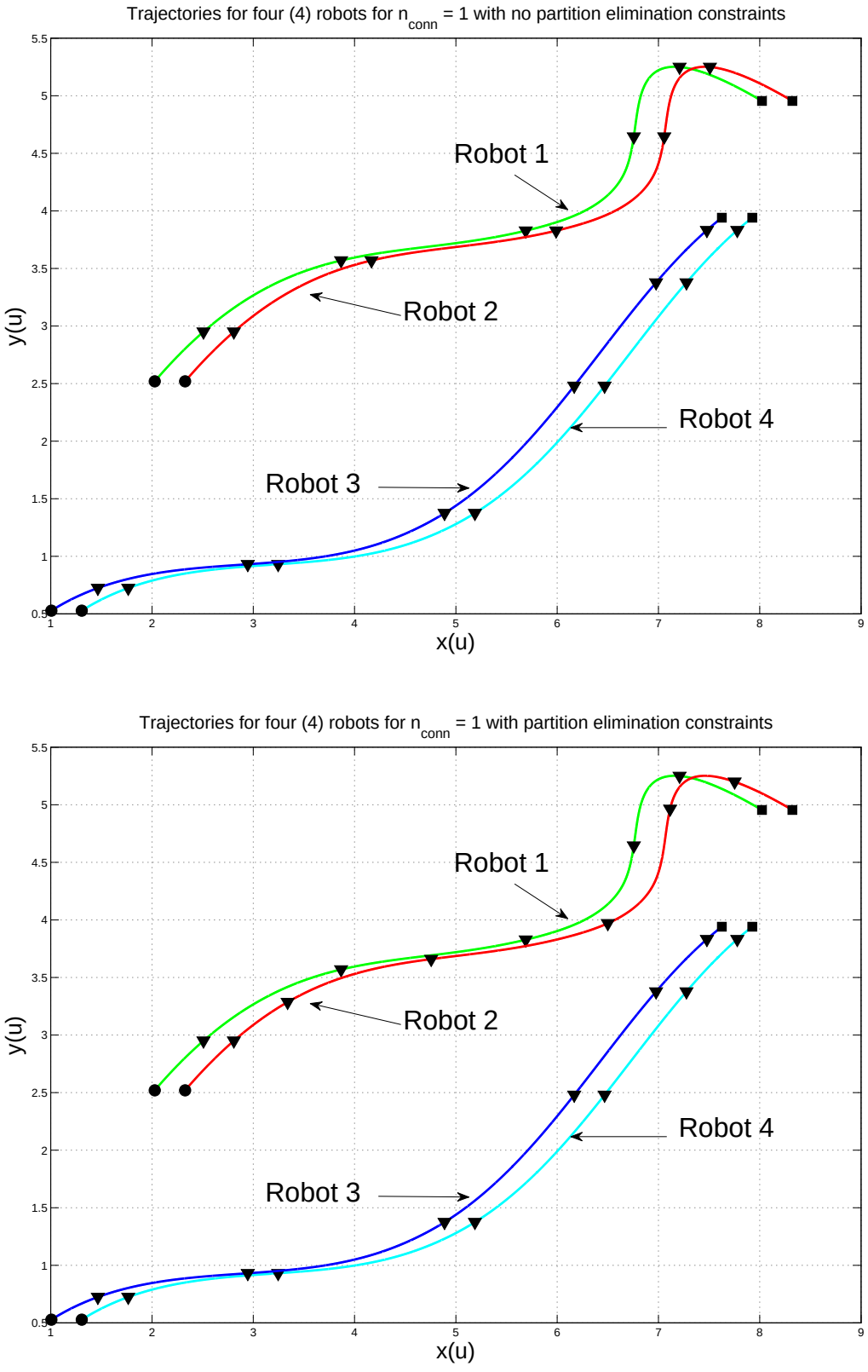

Figure A.7: NLP: A 4 robot scenario with and without the partition elimination constraints

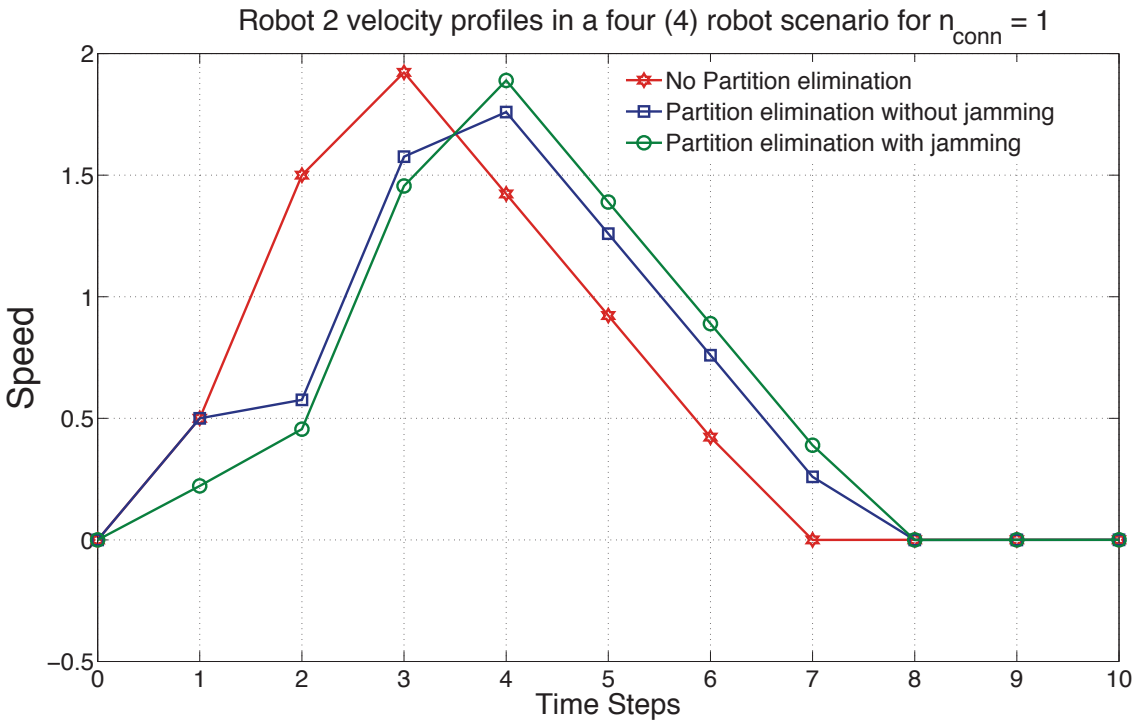

Figure A.8: NLP: Velocity Profile of the robot 2 in absence and presence of the partition elimination constraints and jammer

- *10 robots*:

  Figure A.9 shows the trajectories of the robots in a ten (10) robot scenario in presence (right side plot) and absence (left side plot) of the partition elimination constraints. The velocity profiles of robots 2 and 3 change due to the partition elimination constraints, and are demonstrated in Figure A.10. The transmission power in this case is 1.3 mW and the transmission range is 1.69 meters. Before applying the partition elimination constraints, the value of $T_{\max} = 5$. After the partition elimination constraints were added, the value increases to $T_{\max} = 7$.

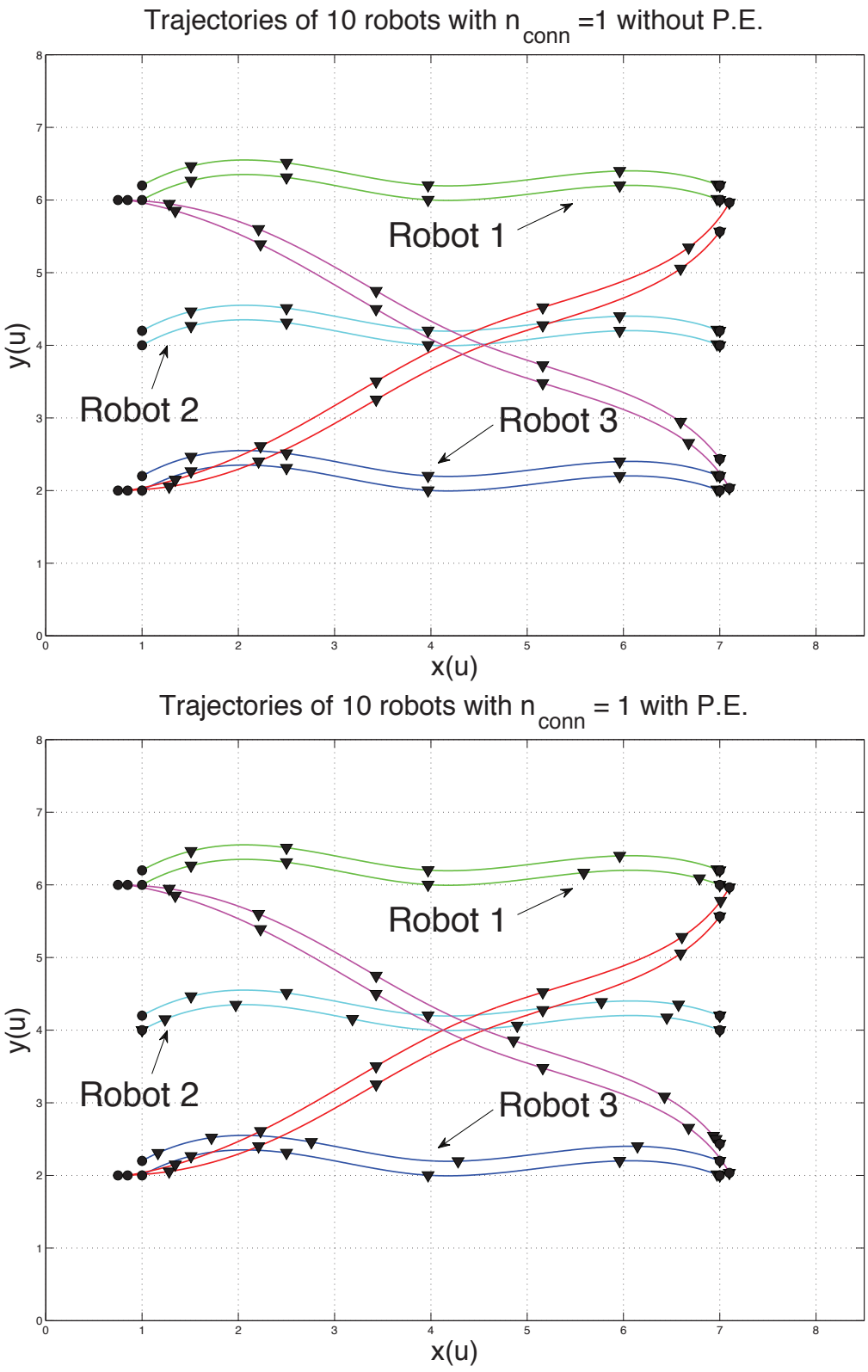

Figure A.9: NLP: A 10 robot scenario with and without the partition elimination constraints

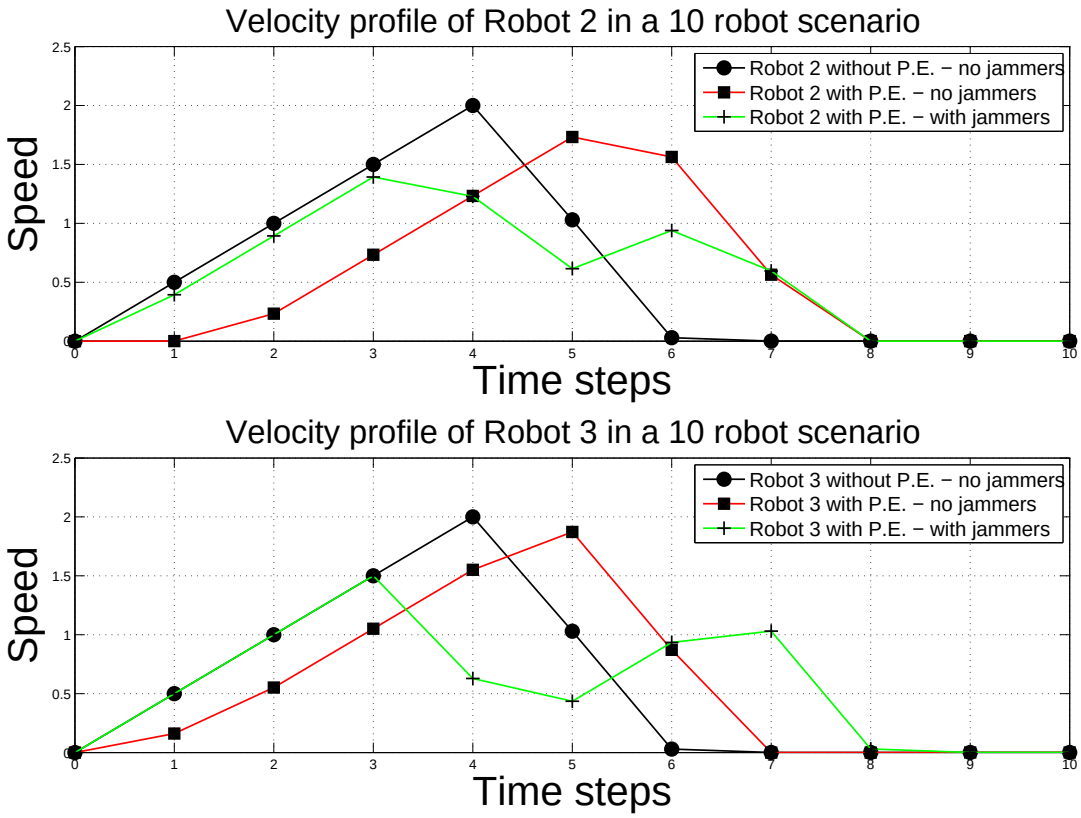

Figure A.10: NLP: Velocity profiles of Robot 2 and Robot 3 in a 10 robot scenario with and without the partition elimination constraints in absence and presence of jammers

- *20 robots*:

  Figure A.11 shows the trajectories of the robots in a twenty (20) robot scenario in presence (right side plot) and absence (left side plot) of the partition elimination constraints. The velocity profile of Robot 1 changes due to the presence of the partition elimination constraints. Since the paths of the robots are very close to each other, the transmission power required for feasibility in this case is 0.2 mW which corresponds to a transmission range of 0.66 meters. Before applying the partition elimination constraints, the value of $T_{\max} = 7$. After the partition elimination constraints were added, the value increases to $T_{\max} = 8$.

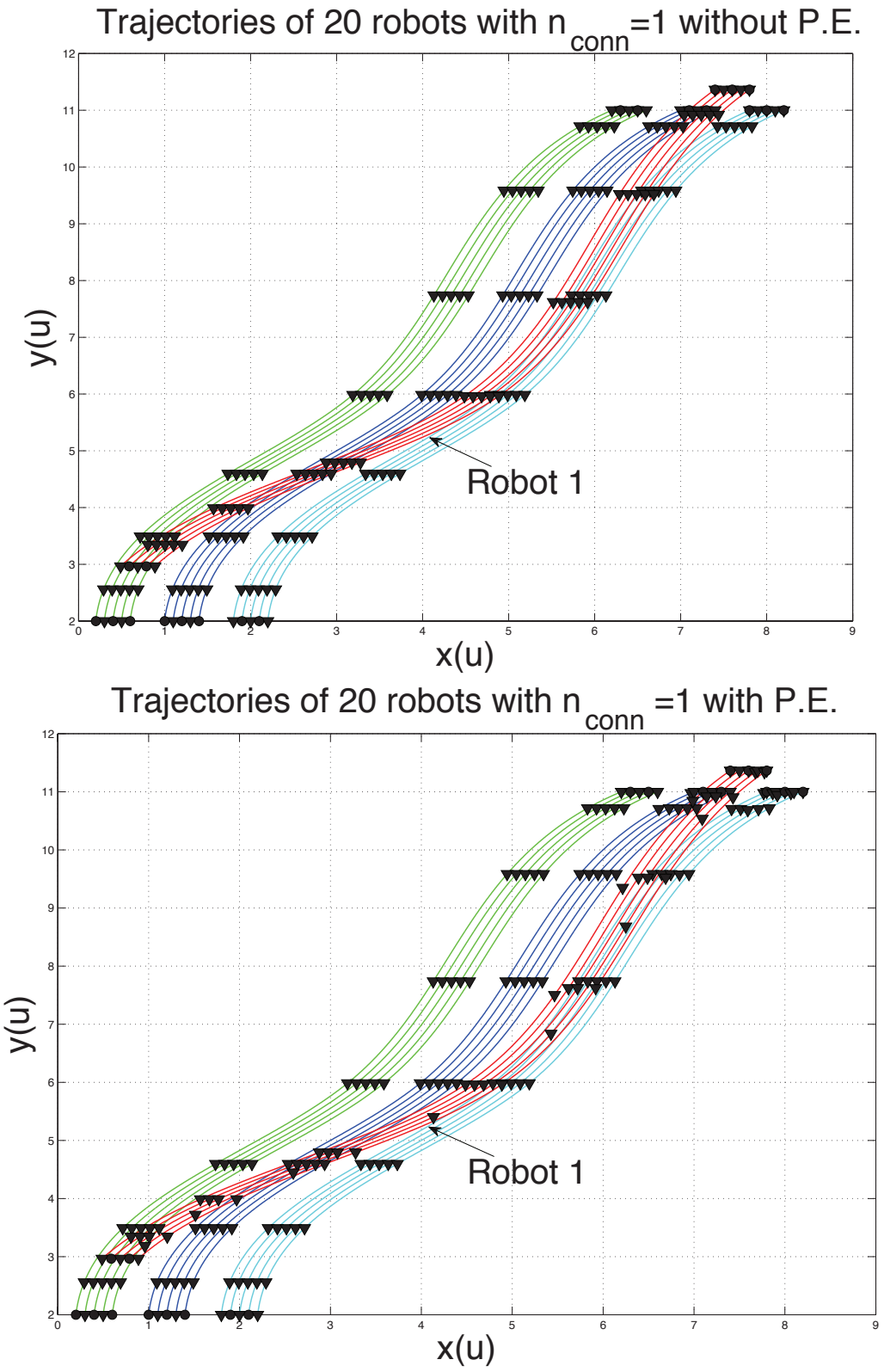

Figure A.11: NLP: A 20 robot scenario with and without the partition elimination constraints

- *50 robots*:

  Figure A.12 shows the trajectories of the robots in a fifty (50) robot scenario in presence (right side plot) and absence (left side plot) of the partition elimination constraints. It can be clearly observed that the robot velocity profiles change considerably in order to comply with the communication requirements. The transmission power in this case is 2.2 mW and the transmission range is 2.2 meters. The value of $T_{\max}$ increases from 7 to 9 after the partition elimination constraints are added.

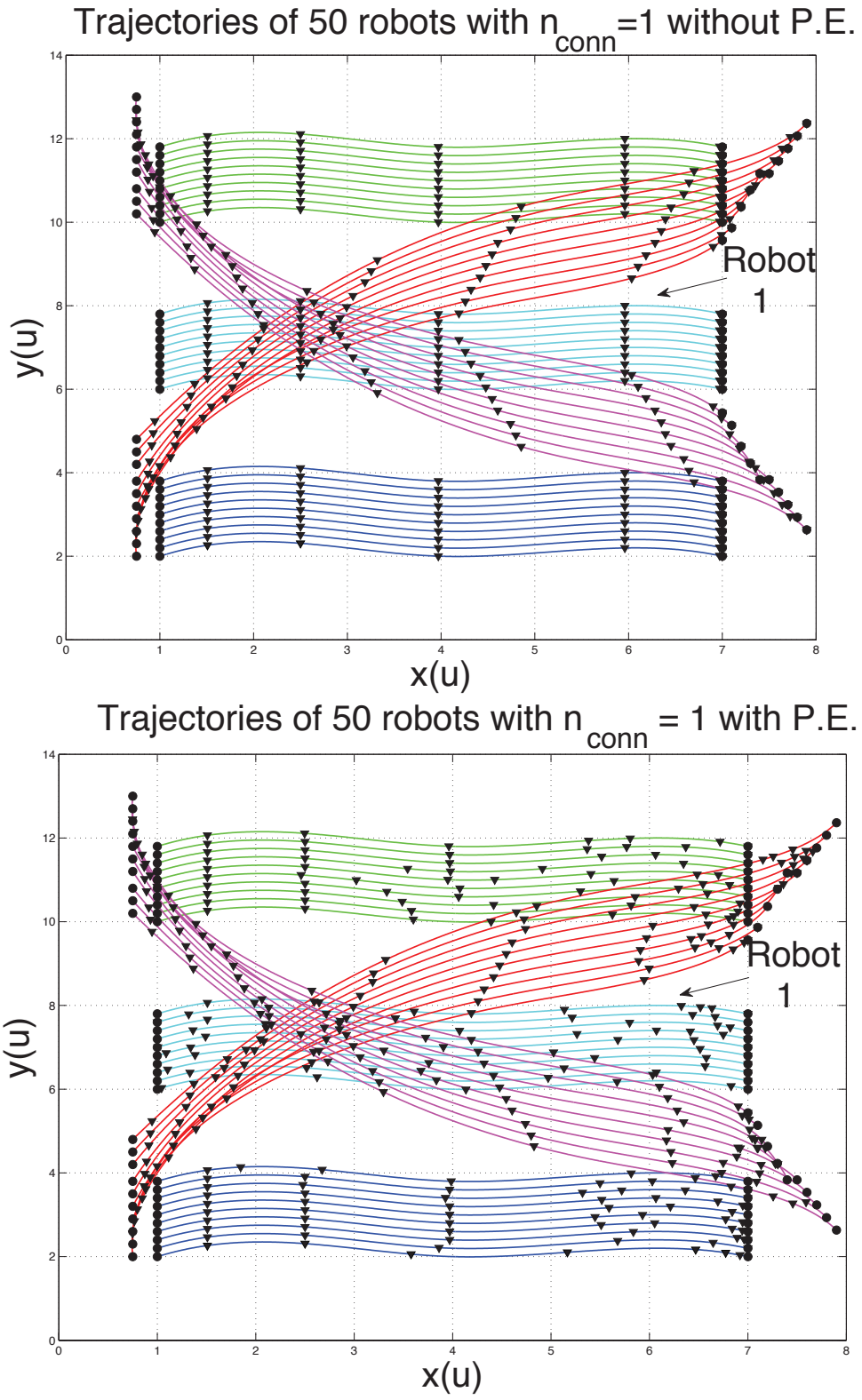

Figure A.12: NLP: A 50 robot scenario with and without the partition elimination constraints

The following observations are made

- The partition elimination constraints make a difference; they make the robots change their speeds in order to maintain connectivity.
- Typically, the "fast" robots whose times of arrival at their respective destinations are less than $T_{\max}$ change their velocity profiles to comply with the new communication connectivity constraint.

Table A.2: NLP: Effect of n on T_{comp} and Problem Size Reduction

n	$T_{\max}$ no PE	$T_{\max}$ with PE	T_{comp}	p no PE algorithm	p with PE algorithm
4	8	8	7.785	60	2
10	5	7	46.869s	6270	3
20	7	8	123.217s	6166450	1
50	7	9	1314.897s	6.2616×10^{15}	1

D. **NLP: Scalability and problem size reduction**

- *Scalability with respect to the number of robots n*:

  We demonstrate the effect of increasing the number of robots n on the computational time. Table A.2 summarizes the results obtained in cases of 4, 10, 20, and 50 robots. The total computational time T_{comp} is measured in seconds and is reported along with the values of $T_{\max}$ for all cases with and without the PE algorithm. It is observed that T_{comp} increases with the increase in the number of robots.

- *Problem size reduction*:

  In our computational experience it is observed that with the use of the P.E. Algorithm in the simulated scenarios, the number of partition elimination constraints p added to the problem are greatly reduced, (see Table A.2).

  For the earlier problem with 4 vehicles, $n_{conn} = 1$, and $T = 10$, we would have had to add 60 partition elimination constraints if we included all such constraints in the problem since there would be $\binom{4}{2}$ possible partitioned subgroups of 2 robots per subgroup at each time period. In this example, by using P.E. algorithm only 2 of these constraints are added, one at time

step 2 and the other at time step 3.

E. **NLP: Effect of presence of jammer robots on the velocity profile**

For a ten robot scenario, Figure A.13 shows trajectories of the robots in absence and presence of a set of 4 jammers. We consider four jammer robots with (black colored) spline paths that move at a constant, predetermined speeds s_{jam}. The triangular markings show the position of the jammers at different time steps. The partition elimination algorithm is applied. When constraints of the form (A.20) are added to the model, the robots change their velocity profiles in order to remain outside the jamming distance of the jammer d_{jam}. The velocity profiles of Robots 2 and 3 are indicated in the Figure A.10. The velocity profiles of these robots are different when compared to the case with no jammer. The value of $T_{\max}$ is 8 in both cases. In this case, $s_{jam} = 0.6$ m/s.

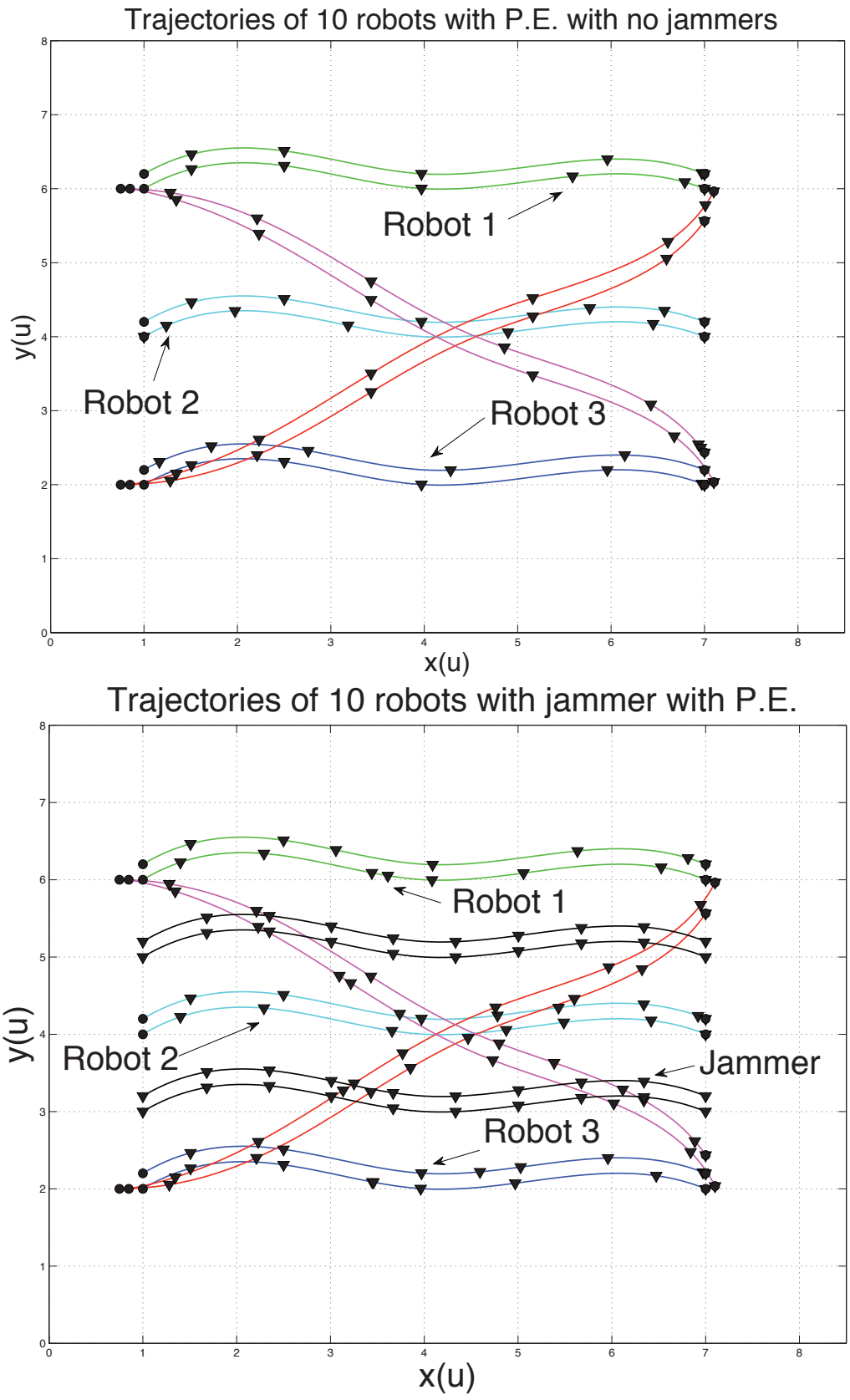

Figure A.13: NLP: A 10 vehicle scenario in absence and presence of 4 jammers traveling at constant speed. The speeds of the robots change in order to avoid being jammed.

Bibliography

[1] J. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, Norwell, MA, 1991.

[2] B. Donald, P. Xavier, J. Canny, and J. Reif, "Kinodynamic Motion Planning," *Journal of the ACM*, vol. 40, no. 5, pp. 1048–1066, 1993.

[3] J. Hopcroft, J. Schwartz, and M. Sharir, "On the complexity of motion planning for multiple independent objects: PSPACE-hardness of the "warehouseman's problem"," *International Journal of Robotic Research*, vol. 3, no. 4, pp. 76–88, 1984.

[4] J. H. Reif, "Complexity of the mover's problem and generalizations," in *Proceedings of IEEE Symposium on Foundations of Computer Science, 1979*, pp. 421–427.

[5] J. Hopcroft, D. Joseph, and S. Whitesides, "Movement problems for 2-dimensional linkages," in *Planning, Geometry, and Complexity of Robot Motion*. Norwood, NJ: Ablex, 1987, pp. 282–329.

[6] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006, available at http://planning.cs.uiuc.edu/.

[7] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press, 2005.

[8] H. Y. Benson, "Mixed Integer Nonlinear Programming using Interior-Point Methods," *Optimization Methods and Software*, to appear.

[9] R. Vanderbei, "LOQO user's manual—version 3.10," *Optimization Methods and Software*, vol. 12, pp. 485–514, 1999.

[10] J. P. Laumond, "Robot Motion Planning and Control," *Lecture Notes in Control and Information Sciences*, vol. 229.

[11] L. Zexiang and J. E. Canny, *Nonholonomic Motion Planning*. Boston, MA, USA: Birkhauser, 1993.

[12] J. Bellingham, M. Tillerson, M. Alighanbari, and J. How, "Cooperative Path Planning for Multiple UAVs in Dynamic and Uncertain Environments," in *Proceedings of the IEEE Conference on Decision and Control*, Dec. 2002, pp. 2816 – 2822.

[13] T. Schouwenaars, J. How, and E. Feron, "Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Providence, RI, August 2004.

[14] N. Yilmaz, C. Evangelinos, P. Lermusiaux, and N. Patrikalakis, "Path Planning of Autonomous Underwater Vehicles for Adaptive Sampling Using Mixed Integer Linear Programming," *IEEE Journal of Oceanic Engineering*, vol. 33, no. 4, pp. 522 –537, oct. 2008.

[15] R. Regele and P. Levi, "Cooperative multi-robot path planning by heuristic priority adjustment," in *Proceeings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2006, pp. 5954 –5959.

[16] S. LaValle and S. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 912 –925, dec 1998.

[17] P. Abichandani, H. Benson, and M. Kam, "Multi-vehicle path coordination under communication constraints," in *Proceedings American Control Conference*, Seattle, WA, June 2008.

[18] ——, "Multi-vehicle path coordination in support of communication," in *Proceedings International Conference on Robotics and Automation, 2009*, Kobe, Japan, May 2009.

[19] J. Peng and S. Akella, "Coordinating multiple robots with kinodynamic constraints along specified paths," *The International Journal of Robotics Research*, vol. 24, no. 4, pp. 295–310, 2005.

[20] P. Abichandani, H. Benson, and M. Kam, "Decentralized path coordination in support of communication," in *Proceedings of International Conference on Robotic Systems*, San Francisco, CA, 2011.

[21] C. Clark, S. M. Rock, and J.-C. Latombe, "Motion planning for multiple mobile robot systems using dynamic networks," in *IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, September 2003, pp. 4222–4227.

[22] N. J. Nilsson, "A mobile automaton: An application of artificial intelligence techniques," in *1st International Conference on Artificial Intelligence*, 1969, pp. 509–520.

[23] J. T. Schwartz and M. Sharir, "On the Piano Movers' Problem: I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers," *Communications on Pure and Applied Mathematics*, vol. 36, pp. 345–398, 1983.

[24] ——, "On the Piano Movers' Problem: II. General techniques for computing topological properties of algebraic manifolds," *Advances in Applied Mathematics*, vol. 12, pp. 298–351, 1983.

[25] T. Lozano-Pérez, "Automatic planning of manipulator transfer movements," *IEEE Transactions on Systems, Man, & Cybernetics*, vol. 11, no. 10, pp. 681–698, 1981.

[26] R. A. Brooks and T. Lozano-Pérez, "A subdivision algorithm in configuration space for findpath with rotation," *IEEE Transactions on Systems, Man, & Cybernetics*, vol. SMC-15, no. 2, pp. 224–233, 1985.

[27] S. Kambhampati and L. Davis, "Multiresolution path planning for mobile robots," *IEEE Journal of Robotics and Automation,*, vol. 2, no. 3, pp. 135 – 145, September 1986.

[28] S. Behnke, "Local multiresolution path planning," in *Proceedings of 7th RoboCup International Symposium*. Springer, 2003, pp. 332–343.

[29] P. Tsiotras, "Multiresolution hierarchical path-planning for small UAVs," in *Proceedings of European Control Conference*, Kos, Greece, July 2007.

[30] C. O'Dunlaing and C. K. Yap, "A retraction method for planning the motion of a disc," *Journal of Algorithms*, vol. 6, pp. 104–111, 1982.

[31] D. Leven and M. Sharir, "Planning a purely translational motion for a convex object in two-dimensional space using generalized Voronoi diagrams," *Discrete and Computational Geometry*, vol. 2, pp. 9–31, 1987.

[32] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[33] C. Warren, "Multiple robot path coordination using artificial potential fields," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 1, 1990, pp. 500 –505.

[34] S. Ge and Y. Cui, "New potential functions for mobile robot path planning," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 5, pp. 615 –620, oct 2000.

[35] D. Koditschek, "Exact robot navigation by means of potential functions: Some topological considerations," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4, March 1987, pp. 1–6.

[36] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Computer Science Dept., Iowa State University, Tech. Rep. 98-11, 1998.

[37] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566 –580, August 1996.

[38] N. Amato and Y. Wu, "A randomized roadmap method for path and manipulation planning," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, 22-28 1996, pp. 113 –120.

[39] J. Bruce and M. Veloso, "Real-time randomized path planning for robot navigation," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2002, pp. 2383 – 2388.

[40] G. A. S. Pereira, A. K. Das, V. Kumar, and M. F. M. Campos, "Decentralized motion planning for multiple robots subject to sensing and communication constraints," in *in Proceedings of the Second MultiRobot Systems Workshop*. Kluwer Academic Press, 2003, pp. 267–278.

[41] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," in *Proceedings of International Conference on Robotics and Automation, 2002.*, vol. 2, 2002, pp. 1217 –1222.

[42] D. Shim, H. J. Kim., and S. Sastry, "Decentralized nonlinear model predictive control of multiple flying robots in dynamic environments," in *Proceedings of IEEE Conference on Decision and Control*, Maui, HI, Dec 2003.

[43] A. Cruz-Martin, V. Munoz, and A. Garcia-Cerezo, "Genetic algorithms based multirobot trajectory planning," in *Proceedings of World Automation Congress, 2004*, vol. 15, june 2004, pp. 155 –160.

[44] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," in *Algorithmic and Computational Robotics: New Directions*. Wellesley, MA: A.K. Peters, 2001.

[45] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, May 2001.

[46] T. Schouwenaars, B. DeMoor, E. Feron, and J. How, "Mixed Integer Programming for Multi-Vehicle Path Planning," in *Proceedings of European Control Conference 2001*, 2001, pp. 2603–2608.

[47] M. Earl and R. D'Andrea, "Iterative MILP methods for vehicle-control problems," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1158 – 1167, Dec. 2005.

[48] C. Reinl and O. von Stryk, "Optimal control of multi-vehicle-systems under comm. constraints using Mixed Integer Linear Programming," in *Proceedings of the 1st International Conference on Robot Communication and Coordination*, Oct. 2007.

[49] G. Inalhan, D. Stipanovic, and C. Tomlin, "Decentralized optimization, with application to multiple aircraft coordination," in *Proceedings of IEEE Conference on Decision and Control*, Las Vegas, NV, Dec 2002.

[50] S. M. LaValle, "A game-theoretic framework for robot motion planning," Ph.D. dissertation, University of Illinois, Urbana, IL, Jul. 1995.

[51] Y. Guo and L. E. Parker, "A distributed and optimal motion planning approach for multiple mobile robots," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2002, pp. 2612–2619.

[52] P. O'Donnell and T. Lozano-Perez, "Deadlock-free and collision-free coordination of two robot manipulators," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, 1989, pp. 484–489.

[53] S. LaValle and S. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 912–925, 1998.

[54] T. Simeon, S. Leroy, and J. Laumond, "Path coordination for multiple mobile robots: a resolution-complete algorithm," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 1, pp. 42–49, 2002.

[55] B. Jung and G. Sukhatme, "Cooperative multi-robot target tracking," in *Proceedings of 8th International Symposium on Distributed Autonomous Robotic Systems*, vol. 1, July 2006, pp. 81–90.

[56] J. Derenick, J. Spletzer, and A. Hsieh, "An optimal approach to collaborative target tracking with performance guarantees," *Journal of Intelligent and Robot Systems*, vol. 56, no. 1-2, pp. 47–68, Sep. 2009.

[57] J. Desai, J. Ostrowski, and V. Kumar, "Controlling formations of multiple mobile robots," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4, May 1998, pp. 2864 –2869.

[58] R. Vidal, O. Shakernia, and S. Sastry, "Formation control of nonholonomic mobile robots with omnidirectional visual servoing and motion segmentation," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, Sept. 2003, pp. 584 – 589.

[59] E. Bicho and S. Monteiro, "Formation control for multiple mobile robots: a non-linear attractor dynamics approach," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, Oct. 2003, pp. 2016 – 2022.

[60] J. Chen, D. Sun, J. Yang, and H. Chen, "Leader-follower formation control of multiple non-holonomic mobile robots incorporating a receding-horizon scheme," *The International Journal of Robotics Research*, vol. 29, no. 6, pp. 727–747, 2010.

[61] M. L. F. Amirouche, *Fundamentals of Multibody Dynamics: Theory and Applications*. Boston, MA, USA: Birkhauser, 2006.

[62] E. Whittaker, *A treatise on the analytical dynamics of particles and rigid bodies: with an introduction to the problem of three bodies*, ser. Cambridge mathematical library. Cambridge University Press, 1988.

[63] R. Fierro and F. L. Lewis, "Control of a nonholomic mobile robot: Backstepping kinematics into dynamics," *Journal of Robotic Systems*, vol. 14, no. 3, pp. 149–163, 1997.

[64] N. Sarkar, X. Yun, and V. Kumar, "Control of mechanical systems with rolling contacts: Applications to mobile robots," *International Journal of Robotics Research*, vol. 13, no. 1, pp. 55–69, Feb. 1994.

[65] N. Sidek and N. Sarkar, "Dynamic modeling and control of nonholonomic mobile robot with lateral slip," in *Third International Conference on Systems*, April 2008, pp. 35 –40.

[66] N. Chakraborty and A. Ghosal, "Dynamic modeling and simulation of a wheeled mobile robot for traversing uneven terrain without slip," *Journal of Mechanical Design*, vol. 127, no. 5, pp. 901–909, 2005.

[67] N. McClamroch and D. Wang, "Feedback stabilization and tracking of constrained robots," *IEEE Transactions on Automatic Control*, vol. 33, no. 5, pp. 419 –426, May 1988.

[68] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[69] J. A. Reeds and L. A. Shepp, "Optimal paths for a car that goes both forwards and backwards." *Pacific Journal of Mathematics*, vol. 145, no. 2, pp. 367–393, 1990.

[70] A. Piazzi, C. Guarino Lo Bianco, and M. Romano, "η^3-splines for the smooth path generation of wheeled mobile robots," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 1089 –1095, oct. 2007.

[71] T. Fraichard and A. Scheuer, "From reeds and shepp's to continuous-curvature paths," *IEEE Transactions on Robotics*, vol. 20, no. 6, pp. 1025 – 1035, dec. 2004.

[72] C. Froissart and P. Mechler, "On line polynomial path planning in cartesian space for robot manipulators," *Robotica*, vol. 11, pp. 245–251, 1993.

[73] P. Gallina and A. Gasparetto, "A technique to analytically formulate and to solve the 2-dimensional constrained trajectory planning problem for a mobile robot," *Journal of Intelligent and Robotic Systems*, vol. 27, pp. 237–262, 2000.

[74] S. Fleury, P. Soueres, J.-P. Laumond, and R. Chatila, "Primitives for smoothing mobile robot trajectories," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 3, pp. 441 –448, Jun 1995.

[75] T. Berglund, A. Brodnik, H. Jonsson, M. Staffanson, and I. Soderkvist, "Planning smooth and obstacle-avoiding b-spline paths for autonomous mining vehicles," *IEEE Transactions on Automation Science and Engineering*, vol. 7, no. 1, pp. 167 –172, Jan 2010.

[76] W. Nelson, "Continuous-curvature paths for autonomous vehicles," in *Proceedings of International Conference on Robotics and Automation*, may 1989, pp. 1260 –1264 vol.3.

[77] ——, "Continuous steering-function control of robot carts," *IEEE Transactions on Industrial Electronics*, vol. 36, no. 3, pp. 330 –337, aug 1989.

[78] A. Takahashi, T. Hongo, Y. Ninomiya, and G. Sugimoto, "Local path planning and motion control for AGV in positioning," in *Proceedings. of IEEE/RSJ International Workshop on Intelligent Robots and Systems*, Sep 1989, pp. 392 –397.

[79] O. Pinchard, A. Liegeois, and F. Pougnet, "Generalized polar polynomials for vehicle path generation with dynamic constraints," in *Proceedings. of IEEE International Conference on Robotics and Automation*, vol. 1, Apr 1996, pp. 915 –920.

[80] Y. Kanayama and B. Hartman, "Smooth local path planning for autonomous vehicles," in *Proceedings. of IEEE International Conference on Robotics and Automation*, May 1989, pp. 1265 –1270 vol.3.

[81] H. Delingette, M. Hebert, and K. Ikeuchi, "Trajectory generation with curvature constraint based on energy minimization," in *Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems*, Nov 1991, pp. 206 –211.

[82] H. Friis, "A note on a simple transmission formula," *In Proceedings of the IRE*, vol. 34, no. 5, pp. 254–256, May 1946.

[83] T. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[84] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, pp. 407–427, 1999.

[85] J. D. Camm, A. S. Raturi, and S. Tsubakitani, "Cutting Big-M down to size," *Interfaces*, vol. 20, no. 5, pp. pp. 61–66, 1990.

[86] R. Fletcher, *Practical Methods of Optimization*, 2nd ed. New York: John Wiley & Sons, 1987, pp. 183–188.

[87] M. Bussieck and V. S., "MINLP solver software," *Accepted for Wiley Encyclopedia of Operations Research and Management Science*. [Online]. Available: http://www.math.hu-berlin.de/~stefan/minlpsoft.pdf

[88] Ilog, Inc., "Solver cplex." [Online]. Available: http://www.ibm.com/software/integration/optimization/cplex-optimizer/

[89] J. F. Sturm, O. Romanko, I. Polik, and T. Terlaky, "Sedumi," 2009, http://mloss.org/software/view/202/.

[90] "The MOSEK optimization software." [Online]. Available: http://www.mosek.com/

[91] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," IBM T. J. Watson Research Center, Yorktown, USA, Tech. Rep. RC 23149, March 2004.

[92] S. Leyffer, "Integrating SQP and branch-and-bound for mixed integer nonlinear programming," *Computational Optimization and Applications*, vol. 18, pp. 295–309, 2001.

[93] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Review*, vol. 47, no. 1, pp. 99–131, 2005.

[94] R. H. Byrd, J. Nocedal, and R. Waltz, "KNITRO: An integrated package for nonlinear optimization," in *Large-Scale Nonlinear Optimization*, G. di Pillo and M. Roma, Eds. Springer, 2006, pp. 35–59.

[95] E. Lawler and D. E. Wood, "Branch-and-bound methods: A survey," *Operations Research*, vol. 15, pp. 699–719, 1966.

[96] R. E. Moore, "Global optimization to prescribed accuracy," *Computers and Mathematics with Applications*, vol. 21(6/7), pp. 25–39, 1991.

[97] P. J. Angeline and K. E. Kinnear, Jr., Eds., *Advances in Genetic Programming - 2*. Cambridge, MA, USA: MIT Press, 1996.

[98] L. Ingber, "Adaptive simulated annealing (ASA)," *Caltech Alumni Association*, 1993.

[99] A. M. Geoffrion, "Generalized benders decomposition," *Journal of Optimization Theory and Applications*, vol. 10, pp. 237–260, 1972.

[100] M. Duran and I. Grossmann, "An outer-approximation algorithm for a class of mixed-integer nonlinear programs," *Mathematical Programming*, vol. 36, pp. 307–339, 1986.

[101] S. Leyffer, "The return of the active set method," *Oberwolfach Reports*, vol. 2(1), 2005.

[102] R. Fourer, D. Gay, and B. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*. Scientific Press, 1993.

[103] A. Brooke, D. Kendrick, and A. Meeraus, *GAMS: A User's Guide*. Scientific Press, 1988.

[104] J. Löfberg, "YALMIP : A toolbox for modeling and optimization in MATLAB," in *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. [Online]. Available: http://users.isy.liu.se/johanl/yalmip

[105] G. Aoude, J. P. How, and I. Garcia, "Two-stage path planning approach for solving multiple spacecraft reconfiguration maneuvers," *AAS Journal. of Astronomical Science*, vol. 56, no. 5, pp. 515 –544, 2008.

[106] A. Richards, Y. Kuwata, and J. How, "Experimental Demonstrations of Real-time MILP Controi," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference 2003*.

[107] T. Schouwenaars, "Safe trajectory planning of autonomous vehicles," *PhD thesis, MIT, 2005*.

[108] J.Peng and S. Akella, "Coordinating multiple double integrator robots on a roadmap: Convexity and global optimality," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2005, pp. 2762–2769.

[109] A. Richards and J. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *Proceedings of American Control Conference*, vol. 3, 2002, pp. 1936 – 1941.

[110] T. Keviczky, F. Borrelli, K. Fregene, D. Godbole, and G. Balas, "Decentralized receding horizon control and coordination of autonomous vehicle formations," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 1, pp. 19 –33, 2008.

[111] T. Keviczky, F. Borrelli, and G. Balas, "A study on decentralized receding horizon control for decoupled systems," in *Proceedings of American Control Conference*, vol. 6, Boston, MA, 2004, pp. 4921 – 4926.

[112] L. Pallottino, E. Feron, and A. Bicchi, "Conflict resolution problems for air traffic management systems solved with mixed integer programming," *IEEE*

Tranasctions on Intelligent Transportation Systems, vol. 3, no. 1, pp. 3 –11, Mar. 2002.

[113] F. Borrelli, D. Subramanian, A. Raghunathan, and L. Biegler, "MILP and NLP techniques for centralized trajectory planning of multiple unmanned air vehicles," in *Proceedings of American Control Conference*, Minneapolis, MN, 2006, p. 6.

[114] L. Singh and J. Fuller, "Trajectory generation for a UAV in urban terrain, using nonlinear MPC," in *Proceedings of American Control Conference*, vol. 3, 2001, pp. 2301 –2308.

[115] I. Garcia and J. How, "Trajectory optimization for satellite reconfiguration maneuvers with position and attitude constraints," in *Proceedings of American Control Conference*, vol. 2, 2005, pp. 889 – 894.

[116] J. Mattingley and S. Boyd, "Automatic code generation for real-time convex optimization," in *Convex Optimization in Signal Processing and Communications*. Eds. Cambridge University Press, 2009.

[117] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267 –278, 2010.

[118] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control*. New York: Hemisphere Publishing Corp., 1975.

[119] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, ser. Studies in Applied Mathematics. Philadelphia, PA: SIAM, Jun. 1994, vol. 15.

[120] J. Gower, "Properties of euclidean and non-euclidean distance matrices," *Linear Algebra and its Applications*, vol. 67, pp. 81–97, June 1985.

[121] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state-dependent graph," *IEEE Transactions on Automatic Control*, vol. 51(1), pp. 116–120, 2006.

[122] R. Ghabcheloo, A. Pascoal, C. Silvestre, and I. Kaminer, "Non-linear coordinated path following control of multiple wheeled robots with bidirectional communication constraints," *International Journal of Adaptive Control and Signal Processing*, vol. 21, no. 2-3, pp. 133–157, 2007.

[123] M. Buehler, K. Iagnemma, and S. Singh, Eds., *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic, George Air Force Base, Victorville, California, USA*, ser. Springer Tracts in Advanced Robotics, vol. 56. Springer, 2009.

[124] M. Lepetic, G. Klancar, I. Skrjanc, D. Matko, and B. Potocnik, "Time optimal path planning considering acceleration limits," *Robotics and Autonomous Systems*, vol. 45, pp. 199–210, 2003.

[125] H. Y. Benson, "Milano: Mixed-integer linear and nonlinear optimizer." [Online]. Available: http://www.pages.drexel.edu/~hvb22/milano/

[126] H. Y. Benson, A. Sen, D. Shanno, and R. J. Vanderbei, "Interior point algorithms, penalty methods and equilibrium problems," *Computational Optimization and Applications*, vol. 34(2), pp. 155–182, June 2006.