Foundations and Trends® in
Robotics
Vol. 2, No. 4 (2011) 261–338

DOI: 10.1561/2300000025

now
the essence of knowledge

Mathematical Programming Approaches for Multi-Vehicle Motion Planning: Linear, Nonlinear, and Mixed Integer Programming

By Pramod Abichandani, Hande Benson and Moshe Kam

Contents

Foundations and Trends® in
Robotics
Vol. 2, No. 4 (2011) 261–338

DOI: 10.1561/2300000025

now
the essence of knowledge

Mathematical Programming Approaches for Multi-Vehicle Motion Planning: Linear, Nonlinear, and Mixed Integer Programming

Pramod Abichandani,[1] Hande Benson[2] and Moshe Kam[3]

[1] *College of Engineering, Drexel University, 3141 Chestnut Street, PA 19104, USA, pva23@drexel.edu*
[2] *College of Business, Drexel University, 3141 Chestnut Street, PA 19104, USA, hvb22@drexel.edu*
[3] *ECE Department, Drexel University, 3141 Chestnut Street, PA 19104, USA, kam@drexel.edu*

Abstract

Real world Multi-Vehicle Motion Planning (MVMP) problems require the optimization of suitable performance measures under an array of complex and challenging constraints involving kinematics, dynamics, collision avoidance, and communication connectivity. The general MVMP problem is thus formulated as a Mathematical Programming (Optimization) problem. In this monograph, we present a Mathematical Programming (MP) framework that captures the salient features of the general MVMP problem. To demonstrate the use of MP for the formulation and solution of MVMP problems, we examine in detail four representative works and summarize several other related ones.

Following this conceptual discussion, we provide a step-by-step demonstration of how to formulate, solve, and experimentally validate an MP problem that represents an MVMP. Finally, we discuss the advantages, technical challenges, and limitations of this framework. As solution algorithms and their implementations in solvers continue to develop, we anticipate that MP solution techniques will be applied to an increasing number of MVMP problems, and that the framework, formulations, and experimental approach presented here may serve as a guide for future MVMP research.

1

The Future is Bright . . . and Driverless

In 2011–2012, Google demonstrated autonomous ground-based mobility in an urban environment. By August 2012, the Google Driverless Car, similar to the one shown in Figure 1.1, had logged more than 300,000 miles in the state of California [134]. Sebastian Thrun, the lead developer of the car described the broader impacts of this technology as follows, "we could change the capacity of highways by a factor of two or three if we didn't rely on human precision on staying in the lane — improve body position and therefore drive a little bit closer together on a little bit narrower lanes, and do away with all traffic jams on highways" [132]. Recently, Newman and others at Oxford University have demonstrated driverless cars with similar capabilities [35]. Several states in the United States have already passed legislation that allow driver's licenses being issued to driverless cars [106].

During the same period, Kumar et al. demonstrated multiple quadrotors operating in an indoor workspace and maintaining formations that translate and rotate with time. This is an example where multiple autonomous vehicles coordinate to achieve a collective task [96].

Fig. 1.1 Google's driverless car had logged more than 300,000 miles in the state of California by August 2012.

These and many other developments are harbingers of a future where multiple autonomous vehicles will become an all pervasive concept with applications that improve our standard of living and lifestyles. Such vehicles will lead to a reduction in the number of accidents and commute times, and improved fuel efficiency. These, in turn, will drive down the cost of public transportation, logistics, and supply chain management, thereby allowing transportation of personnel and goods to previously unreachable locations in record times.

While exciting, such autonomous vehicle-based applications come with their own challenges. For example, vehicles may need to plan their motions in real-time while in transit. This necessitates that the autonomous decision making be dynamic and efficient. There are kinematic challenges as well, e.g., cars cannot slide sideways to parallel park and a fixed-wing aircraft has a nonzero turn radius. Design limitations such as finite battery and fuel capacity add to the complexity of the situation.

Furthermore, if multiple vehicles are operating in a common workspace (e.g., urban environments, highways, airways), some shared knowledge of their behavior is important. To obtain this knowledge, either one needs to have sensors or the vehicles need to communicate

with each other or with central entities that update other vehicles with the latest information about their intentions as and when needed.

The biggest challenges, however, will be the commercial viability of such vehicles and adoption by the general public. Specifically, we need to address issues of energy efficiency and safety.

- Efficiency requirements dictate that these vehicles use as little fuel as possible, to travel as far as possible, as quickly as possible.
- Safety requirements dictate that these vehicles be capable of avoiding obstacles and not collide with each other.

Despite all these challenges, the current propensity of innovation in this space points to a future that is indeed bright...and driverless!

1.1 What's in it for the Readers?

In the following discussions, the readers will be introduced to a rigorous and systematic treatment of these challenges and a family of technical approaches that will assist in addressing them. Specifically, we will ground our discussions in the idea of mathematical programming as applied to Multiple-Vehicle Motion Planning (MVMP) and present the material in the following order:

- In Section 2, we start by formally defining the MVMP problem and its most common variants, such as path planning, trajectory planning, and path coordination problems. We then present the basic elements for modeling MVMP systems in detail. These elements include vehicle kinematics, dynamics, path primitives, and communication models. We also provide an introduction to the mathematical programming framework that will be used to model MVMP problems, relevant solution algorithms, modeling environments, and solvers.
- A general MP based framework that captures the salient features of MVMP problems is introduced in Section 3. We start

this discussion with a review of existing literature of MVMP using MP by focusing on four representative papers that best demonstrate the application of this framework to a range of MVMP scenarios. Each of the four papers provides various model components, which are presented in great detail. Key analyses performed in these papers have been independently derived and presented in the Appendices.

- In Section 4, readers are provided a step-by-step demonstration on how to formulate and solve a distributed MVMP involving autonomous unmanned ground vehicles operating in an indoor environment under communication connectivity constraints. Experimental approaches utilized in the literature to validate MP based MVMP formulations are documented.
- Finally, in Section 5, we discuss the technical challenges and limitations of this framework and present future directions of this research.

2

The Building Blocks

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* [80]. This problem is also called the *kinodynamic motion planning problem* [45], because it takes into account the kinematics, dynamics, and collision avoidance requirements of the robot. The situation becomes much more difficult to formulate and solve once extended to the coordinated motion of multiple mobile robots amidst multiple stationary and moving obstacles [63, 64, 115].

Figure 2.1 is an example of a general motion planning scenario involving multiple robotic vehicles with given initial positions and goal locations. The environment also consists of static obstacles. Figure 2.2 illustrates a set of possible solution trajectories for the robots. The fact that these robotic vehicles share a common workspace raises several

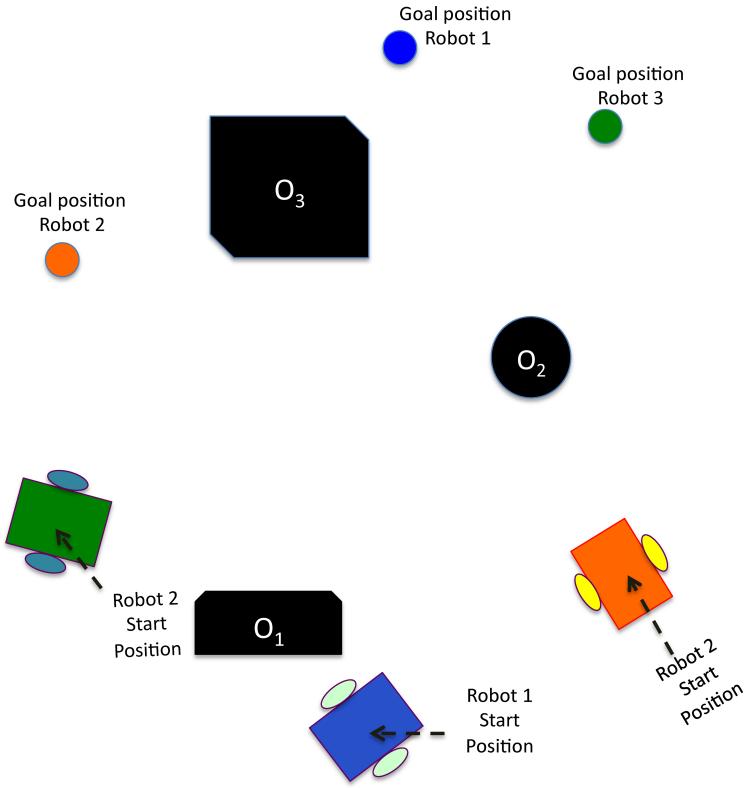


Fig. 2.1 Trajectory planning problem: robots in their workspace; black blocks indicate obstacles.

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 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 [80] was published less than two decades ago and to date, only a handful of textbooks cover it in detail [37, 85]. Coordinated motion planning of multiple robotic

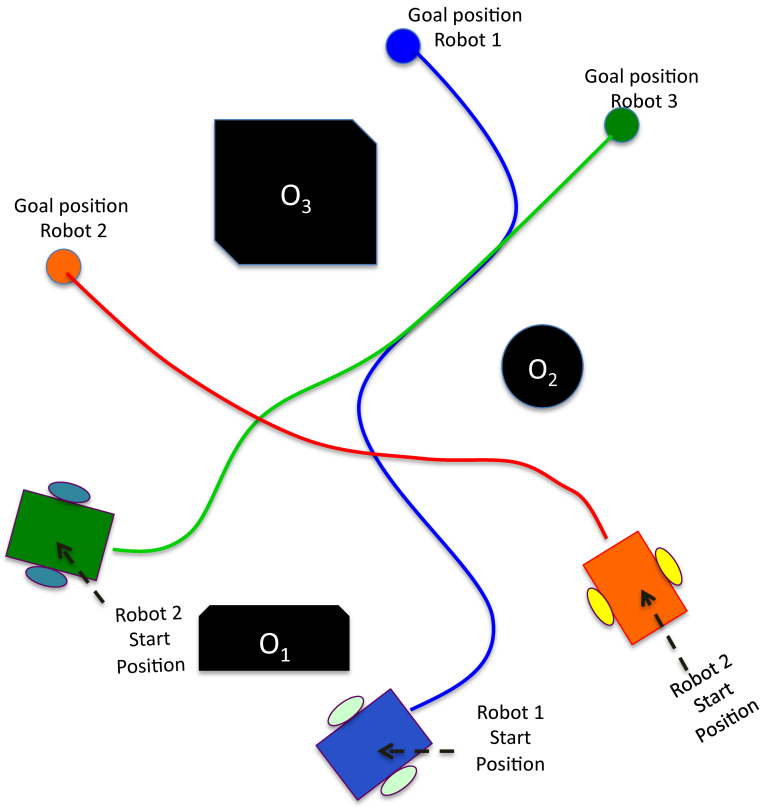


Fig. 2.2 Solution trajectories that satisfy kinematics, dynamics, and collision avoidance.

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.

2.1 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 [103], exact cell decomposition [123, 124] and approximate cell decomposition [18, 28, 71, 92, 133], Voronoi diagrams [88, 105], potential fields [58, 76, 142], navigation functions [79], and randomized path planning [14, 29, 73, 84] 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 [110, 125, 129], genetic algorithms [40], randomized planners [65, 86], mathematical programming based planners [48, 116, 121, 122], iterative bargaining schemes [67], game-theoretic frameworks [83], dynamic sensing and communication networks [38], column generation [44], and search algorithms [62] 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 [104], constrained configuration space roadmap [82], and grouping robots with shared collision zones into subgroups [127]. In [109], 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 [7, 8, 9], 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* [42, 70] and *Formation Control* [24, 36, 43, 138] are realized by placing additional constraints on one or more of the above-mentioned variants.

2.2 Elements of MVMP Problems

To model an MVMP problem effectively, motion constraints 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. Finally, the inter-robot wireless communication models must consider the stochastic nature of wireless communication. We will now thoroughly discuss each of these elements.

2.2.1 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 [15]. 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 [81] and are usually written as

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

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 Equations (2.2)–(2.4). The associated non-holonomic constraint that prevents the robot from

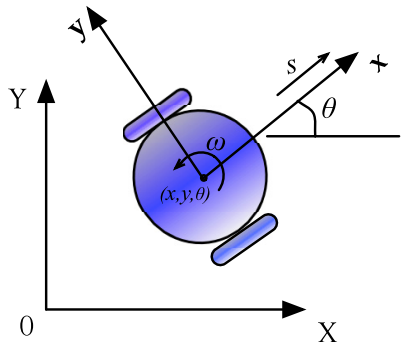


Fig. 2.3 Two-wheeled differential drive robot vehicle — x, y and θ are the generalized coordinates.

sliding sideways is expressed as (2.5).

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

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

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

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

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.

2.2.2 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 (2.7) features Lagrangian function $\mathcal{L}$ which is defined by (2.6) as the difference between the Kinetic

Energy $\mathcal{V}$ and Potential Energy $\mathcal{U}$ of a system [143].

$$\mathcal{L}(q,\dot{q}) = \mathcal{V} - \mathcal{U}(q) \tag{2.6}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\partial q_i} = \mathcal{F}_i + \Psi_i \tag{2.7}$$

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 ith 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 are described by

$$M(q)\ddot{q} + F(q,\dot{q})\dot{q} + K(\dot{q}) + G(q) + \chi = B(q)d + C(q)^T\lambda. \tag{2.8}$$

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) [34, 49, 94, 119, 126].

2.2.3 Path Primitives

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

$$\mathcal{P}:[\varpi_0,\varpi_1] \to \mathbb{R}^2, \quad \varpi \to \mathcal{P}(\varpi) = [\alpha(\varpi)\beta(\varpi)]^T \tag{2.9}$$

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

The arc length u along the path is evaluated by the relationship

$$u = \int_{\varpi_0}^{\varpi_1} \|\dot{\mathbf{p}}(z)\|\mathrm{d}z. \tag{2.10}$$

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

The seminal results on the shortest paths for car-like robots by Dubins [46] and Reeds and Shepp [114] use straight lines and circular arcs as path primitives. The optimality (shortest path) 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 [111]. Furthermore to avoid slipping, the curvature should be continuously differentiable (G^3 path) [111]. 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, and intrinsic spline are parametric curves whose curvature is a function of their arc length [53]. Table 2.1 summarizes different primitives that have been used in the literature.

2.2.4 Inter-Robot Communication

Assuming that each vehicle is equipped with a wireless transceiver, the signal to noise ratio (SNR) experienced by the receiver robot is used to determine whether or not the communication is successful. 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 with each other. As we will see momentarily, if the transmit power and environmental conditions are fixed, the received power level is determined uniquely by the distance between the robots. In addition, the probability of communication success under certain

[1] The curvature should be less than 1 unit for Reeds and Shepp, and Dubins cars [81].

Table 2.1. Path primitives used to represent robot paths.

Name	Path primitives	Features
Dubins [46]	Straight lines and circular arcs	Minimum curvature paths, discontinuity in curvature
Reeds and Shepp [114]	Straight lines and circular arcs	Minimum curvature paths with cusps allowed
Froissart and Mechler [55]	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 [56]	Parametric sum of harmonics	Sums of sine and cosine functions, continuous derivates of any order, closed form expressions available
Fleury et al. [51]	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 [56]
Berglund et al. [23]	B-splines	Smooth closed form splines, quartic B-splines can be designed to have continuous derivatives of curvatures
Nelson [100, 101]; Takahashi et al. [131]; Pinchard et al. [112]	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 [72]	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. [41]	Intrinsic spline	Curves with polynomial curvature profile
Fraichard and Scheuer [53]	CC steer method	Paths comprised of straight lines, circular arcs, and clothoids; have continuous bounded curvature and bounded curvature derviative
Piazzi et al. [111]	η^2 and η^3 Splines	Unified framework to generate or approximate a variety of curve primitives such as circular arcs, clothoids, spirals, etc.

types of uncertainty are dependent on this distance. Therefore, if the SNR exceeds η_c, we also say that the two robots are in 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} and is calculated using Friis's equation[54]

$$P_{\mathrm{r}}^{j}(d^{ij}) = P_{\mathrm{t}}^{i} G_{\mathrm{t}} G_{\mathrm{r}} \left(\frac{\lambda}{4\pi d^{ij}} \right)^{\alpha} \tag{2.11}$$

here, 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 $c = 3 \times 10^{8}\,\mathrm{m/s}$ and $f = 2.4 \times 10^{9}\,\mathrm{Hz}$. The values of G_{t} and G_{r} (antenna gains) are assumed to be 1.

Using (2.11), the average large-scale path loss between a transmitter–receiver pair is expressed as a function of distance:

$$\overline{\mathrm{PL}}^{ij}(d^{ij}) \propto \left(\frac{d^{ij}}{d_0} \right)^{\alpha} \tag{2.12}$$

When expressed in decibels (dB), (2.12) is rewritten as

$$\overline{\mathrm{PL}}^{ij}(d^{ij}) = \overline{\mathrm{PL}}^{ij}(d_0) + 10\alpha \log \left(\frac{d^{ij}}{d_0} \right) \tag{2.13}$$

In (2.12) and (2.13), 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 (2.11) such that $d \geq d_0 \geq d_{\mathrm{f}}$, where d_{f} is the far-field distance. d_{f} is calculated as

$$d_{\mathrm{f}} = \frac{2D_{\mathrm{dim}}^{2}}{\lambda} \tag{2.14}$$

where D_{dim} is the largest physical linear dimension of the antenna. $d_{\mathrm{f}} \gg D_{\mathrm{dim}}$ and $d_{\mathrm{f}} \gg \lambda$ for far-field regions. The bars in (2.12) and (2.13) denote the ensemble average of all possible path loss values for a given value of d.

In addition to the path-loss, shadowing effects due to the 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 (2.12) as indicated by

$$\overline{\mathrm{PL}}^{ij}(d^{ij}) \propto \left(\frac{d^{ij}}{d_0} \right)^{\alpha} \Upsilon \tag{2.15}$$

where Υ has a pdf

$$f_{\Upsilon}(x) = \frac{10/\ln 10}{\sqrt{2\pi}\upsilon x} \exp\left\{-\frac{(10\log_{10} x)^2}{2\upsilon^2}\right\}, \quad x \geq 0. \tag{2.16}$$

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

$$\mathrm{PL}^{ij}(d^{ij})[\mathrm{dB}] = \overline{\mathrm{PL}}^{ij}(d_0) + 10\alpha \log\left(\frac{d}{d_0}\right) + \Upsilon_{\upsilon} \tag{2.17}$$

where Υ_{υ} is the zero-mean Gaussian random variable with standard deviation υ. The received power $P_{\mathrm{r}}^{j}(d)$ is expressed as

$$P_{\mathrm{r}}^{j}(d^{ij})[\mathrm{dBm}] = P_{\mathrm{t}}^{i}[\mathrm{dBm}] - \mathrm{PL}^{ij}(d^{ij})[\mathrm{dB}]. \tag{2.18}$$

The SNR experienced by the receiver robot is calculated using the relationship

$$\mathrm{SNR}^{ij} = P_{\mathrm{r}}^{j}(d^{ij})/N \tag{2.19}$$

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

The *outage probability* $\mathbb{P}_0^{ij}$ is defined as the probability that the SNR experienced is less than a certain threshold η_{SNR}, or equivalently the received power P_{r}^{j} is below a certain threshold η_{c}:

$$\mathbb{P}_0^{ij} = \mathbb{P}(\mathrm{SNR}^{ij} < \eta_{\mathrm{SNR}}) = \mathbb{P}(P_{\mathrm{r}}^{ij} < \eta_{\mathrm{c}}) \tag{2.20}$$

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.

$$\begin{aligned} & \mathbb{P}_0^{ij} \leq \eta_{\epsilon} \\ \Longleftrightarrow\ & \mathbb{P}(\mathrm{SNR}^{ij} < \eta_{\mathrm{SNR}}) \leq \eta_{\epsilon} \\ \Longleftrightarrow\ & \mathbb{P}(P_{\mathrm{r}}^{j}(d^{ij}) < \eta_{\mathrm{c}}) \leq \eta_{\epsilon} \end{aligned} \tag{2.21}$$

Interested readers are referred to [113] and its references for further information about physical layer communication models.

The probabilistic constraint in (2.21) represents inter-vehicle communication connectivity. Recent results in [10] prove that this constraint is transformed into an equivalent constraint on inter-vehicle communication range as follows:

$$\mathbb{P}(\mathrm{SNR}^{ij} < \eta_{\mathrm{SNR}}) = \mathbb{P}(P_{\mathrm{r}}^{j}(d^{ij}) < \eta_{\mathrm{c}}) \tag{2.22}$$

$$= Q\left(\frac{\overline{P_{\mathrm{r}}(d^{ij})} - \eta_{\mathrm{c}}}{\upsilon}\right) \tag{2.23}$$

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

$$Q(z) = \frac{1}{\sqrt{2\pi}} \int_{z}^{\infty} \mathrm{e}^{-x^2/2}\, \mathrm{d}x. \tag{2.24}$$

For a given outage probability threshold for the transmitter–receiver robot pair $i-j$, η_{ε}, the condition $\mathbb{P}_0^{ij} \leq \eta_{\varepsilon}$ leads to the following:

$$\mathbb{P}_0^{ij} \leq \eta_{\varepsilon} \Longleftrightarrow \mathbb{P}(P_{\mathrm{r}}^{j}(d^{ij}) < \eta_{\mathrm{c}}) \leq \eta_{\varepsilon} \tag{2.25}$$

$$\Longleftrightarrow Q\left(\frac{\overline{P_{\mathrm{r}}(d^{ij})} - \eta_{\mathrm{c}}}{\upsilon}\right) \leq \eta_{\varepsilon}$$

$$\Longleftrightarrow \overline{P_{\mathrm{r}}(d^{ij})} \leq \eta_{\mathrm{c}} + \upsilon Q^{-1}(\eta_{\varepsilon})$$

$$\Longleftrightarrow P_t^i[dBm] - \overline{PL}^i j(d_0) - 10\alpha \log \frac{d^{ij}}{d_0} \leq \eta_{\mathrm{c}} + \upsilon Q^{-1}(\eta_{\varepsilon})$$

$$\vdots$$

$$d^{ij} \leq 10\hat{}\left(\frac{\eta_{\mathrm{c}} + \upsilon Q^{-1}(\eta_{\varepsilon}) - P_{\mathrm{t}}^{i}[\mathrm{dBm}] + \overline{\mathrm{PL}}^{ij}(d_0) - 10\alpha \log(d_0)}{-10\alpha}\right) \tag{2.26}$$

Thus, the condition $\mathbb{P}_0^{ij} \leq \eta_{\varepsilon}$ leads to the condition $d^{ij} \leq \eta_{\mathrm{rd}}$, where η_{rd} is given by right-hand side of the inequality (2.26). It is clear that the distance required by this robust constraint is smaller than the deterministic constraint. Based on this formulation, we conclude that the one-hop communication threshold should be reduced in order to decrease the outage probability.

2.3 Mathematical Programming

In its most general form, a mathematical program (2.27) 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 $\Omega : \mathbb{R}^{n_1} \times \mathbb{Z}^{n_2} \to \mathbb{R}^m$

$$\begin{aligned} \text{minimize} \quad & \Phi(\zeta, \sigma) \\ \text{subject to} \quad & \Omega(\zeta, \sigma) \leq 0, \quad q = 1, \ldots, m \\ & \zeta \in \mathbb{R}^{n_1}, \sigma \in \mathbb{Z}^{n_2} \end{aligned} \tag{2.27}$$

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

2.3.1 Relevant MP Classes

The particular form of the problem and the choice of a solution method depends on n_1 and n_2, the numbers of continuous and discrete variables, respectively, as well as the general characteristics of the functions Φ and Ω. When $n_1 = 0$, we have a pure integer optimization problem; when $n_2 = 0$, we have a continuous optimization problem; and when $n_1, n_2 > 0$, we have a mixed-integer optimization problem. A nonlinearity in Φ or any element of Ω leads to the whole problem being referred to as nonlinear. Therefore, for an optimization problem to be classified as linear, Φ and all elements of Ω have to be linear or affine. The three problem classes we will consider are:

- *Mixed-integer linear programming (MILP)*: Φ and all elements of Ω are linear and $n_1, n_2 > 0$.
- *Nonlinear programming (NLP)*: at least one nonlinear term among Φ and Ω, $n_1 > 0$, and $n_2 = 0$.
- *Mixed-integer nonlinear programming (MINLP)*: at least one nonlinear term among Φ and Ω and $n_1, n_2 > 0$. Continuous relaxations obtained by removing the constraints $\sigma \in \mathbb{Z}^{n_2}$ are NLPs.

MILP is a widely studied class of problems [102], and the underlying linear structure lends itself well to obtaining favorable theoretical guarantees and efficient computational performance. As such, many

problems that may be more naturally formulated as MINLPs are reformulated (by linearization) or relaxed (by removing nonlinear terms) as MILPs, which may help in quickly obtaining approximate solutions.

Similarly, NLP is also a widely studied problem class [99]. However, the nonlinearities in the problem may make the guaranteeing of solutions and the development and analysis of efficient algorithms quite complicated. Standard optimality conditions for NLPs require all functions Φ and Ω to be twice continuously differentiable. In addition, most NLP methods are designed to find local optima, and, therefore, one must rely on problem characteristics to identify whether a local optimum is also a global one. If Φ is a convex function and the feasible region of the problem is a convex set, then any local optimum is also a global optimum. However, many NLPs, including those studied here, are nonconvex. In that case, globalization strategies are used to overcome negative curvature and search for a minimum, rather than a first-order point. Nevertheless, such globalization strategies only serve to locate a local optimum "globally," i.e., from any initial solution, so their success at finding a global optimum depends on the quality of the initial solution. Global optimization algorithms and heuristics for NLP do exist, but their applicability and efficiency are still significantly limited by problem size and the types of nonlinear components in Φ and/or Ω.

The most general of these problem classes, MINLP, is also the newest in terms of the development of theory and solution algorithms. It combines the challenges of an NLP with the difficulty of handling discrete variables. Due to the presence of these discrete variables, the feasible region of an MINLP is always disconnected, and, therefore, an MINLP is always a nonconvex optimization problem. Solution approaches for an MINLP seek to solve a series of NLP relaxations, which means that the solution of each NLP relaxation needs to identify a global optimum and do so efficiently. As such, the convexity of the NLP relaxation is highly desirable, as are solution algorithms capable of warmstarts from previous solutions in the sequence of NLPs solved. The models that we discuss in Section 4 are broadly classified as MINLPs 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 [12].

It is also important to note that in experimenting with problem formulations, it may be a good idea to consider reformulations of the objective function or some constraints in order to obtain a version of the model with more favorable properties. One such reformulation is that of an *either–or* (disjunctive) constraint. While disjunctive constraints describe a disconnected feasible region using nondifferentiable functions, we can introduce auxiliary binary variables and convert them to Big-M constraints [19], which still define a nonconvex set but do so using continuous constraint functions. An example of a Big-M constraint is provided later in this discussion.

2.3.2 Solution Algorithms

The difficulty of solving (2.27) 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 [31, 50]. Commercial and/or open-source solvers such as `CPLEX` [66], `SeDuMi` [130], `MOSEK` [4], `LOQO` [137], `IPOPT` [140], `MINLP` [89], `SNOPT` [60], `GUROBI` [3], `SCIP` [5], `GLPK` [2], `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 [50]. Finding globally optimal solutions for non-convex problems can be challenging. Non-convexities are introduced in the problem (2.27) 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 are able to handle non-convexities but do not provide guarantees of finding globally optimal solution. These include solvers such as `LOQO` that uses interior-point methods [137], `KNITRO` that uses trust-region algorithms [32], `SNOPT` that uses a quasi-Newton algorithm [60], and `Bonmin` [1] among others. These solvers also incorporate mechanisms to detect infeasibilities and unboundedness in the problem. A variety of techniques such as branch-and-bound [87, 98], evolutionary algorithms [16], and adaptive simulated annealing (ASA) [69] 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 [59], and outer approximation [47] to handle the integer variables, and active set methods [90] or interior-point methods to handle the continuous relaxations.

2.3.3 Modeling Environments

Modeling environments such as `AMPL` [52], `GAMS` [27], `MATLAB`, and `YALMIP` [91] allow for quick development of models for testing/ simulation purposes. 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.

2.3.3.1 Big-M constraints for disjunctive constraints

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

$$\mathcal{V}_{\mathrm{c}} - M * \mathcal{V}_{\mathrm{d}} \leq 0 \tag{2.28}$$

where M is a large number.

It is clear that when $\mathcal{V}_{\mathrm{d}} = 0$, (2.28) gives $\mathcal{V}_{\mathrm{c}} \leq 0$, which along with the constraint that $\mathcal{V}_{\mathrm{c}} \geq 0$ results in $\mathcal{V}_{\mathrm{c}} = 0$. When $\mathcal{V}_{\mathrm{d}} = 1$, (2.28) gives $\mathcal{V}_{\mathrm{c}} \leq M$ 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 the M value. Selecting a sufficiently large value for M is necessary for the constraints to be trivially satisfied. However, larger values lead to numerical instability in the solution process [33]. Other methods to work with disjunctive constraints include the Balas' disjunctive union method [39].

3

General Mathematical Programming Structure of Multi-Vehicle Motion Planning problems

The body of work that uses MP techniques to address variants of the general MVMP problem continues to grow [11].

3.1 Representative Works

Table 3.1 provides a short introduction to several 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 works, we have selected four for a deeper look. These are studies by Schouwenaars et al. [122], Peng and Akella [109], Derenick et al. [42], and Abichandani et al. [8]. Collectively, these four works cover almost all of the features of MVMP problems, viz. kinematics, dynamics, collision avoidance, communication, and visibility/tracking.

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	$\mathcal{O}$ and Features
Schouwenaars et al. [120, 122]	Discrete time, decentralized trajectory planning of multiple UAVs with safety guarantee	✓	✓	✓	—	Hard safety at all times MILP, `CPLEX`
Peng and Akella [108, 109]	Discrete space, continuous time centralized Kinodynamic fixed path coordination of multiple robots	✓	✓	✓	—	MINLP, MILP approximations, `CPLEX`
Derenick et al. [42]	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. [7, 8, 9]	Discrete time, centralized and decentralized muti-vehicle path coordination under communication constraints	✓	✓	✓	✓	Connected communication graph at all times MINLP, NLP, `LOQO`, `MILANO`, AMPL, MATLAB
Inalhan et al. [68]	Discrete time, trajectory planning of multiple UAVs via optimal bargaining process	✓	✓	✓	✓	NLP, `fmincon`, MATLAB
Richards and How [117]	Discrete time, centralized trajectory planning of multiple UAVs	✓	✓	✓	—	MILP, `CPLEX`, AMPL, MATLAB
Keviczky et al. [74, 75]	Discrete time, decentralized receding horizon control and coordination of autonomous vehicle formations	✓	✓	✓	✓	MILP, `CPLEX`, AMPL, MATLAB
Pallottino et al. [107]	Discrete time, conflict resolution for multiple aircrafts in a shared airspace	—	✓	✓	—	MIP, `CPLEX`
Borrelli et al. [25]	Discrete time, centralized trajectory generation of multiple UAVs	✓	✓	✓	—	MILP, NLP `CPLEX`, `IPOPT`
Singh and Fuller [128]	Discrete time, centralized trajectory generation of a single UAV, extendable to multiple UAVs	✓	✓	—	—	QP, `quad- prog`, MATLAB

(*Continued*)

Table 3.1. (*Continued*)

Name	MVMP variant	$\mathcal{K}$	$\mathcal{D}$	$\mathcal{C}$	Θ	$\mathcal{O}$ and Features
Aoude et al. [17]	Discrete time, multiple spacecraft reconfiguration Maneuvers	✓	✓	✓	—	Pointing restrictions NLP, `SNOPT`
Mellinger et al. [95]	Discrete time, trajectory generation for heterogeneous quadrotor teams	✓	✓	✓	—	MIQP, `CPLEX`
Yilmaz et al. [144]	Path planning of autonomous underwater vehicles	✓	✓	✓	—	MILP, `XPress - MP, Mosel` [6]
van den Berg et al. [136]	Reciprocal velocity obstacles for multi-agent navigation	—	—	✓	—	LP, `RVO` [136]

As most MVMP problems, the problems studied in [8, 42, 122, 109], are expressed using the following mathematical program formulation:

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

Schouwenaars et al. [122] present a cooperative decentralized algorithm for trajectory generation of multiple UAVs with hard safety guarantees. Of main interest in this monograph is the Mixed Integer Linear Programming (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. This paper led to one of the first practical implementations of MP based motion planning of UAVs [120].

In [109], 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 to obtain closed form formulae for minimum and maximum times that the double integrator should take to traverse a path segment of given length. Specifically, the authors make use of minimum and maximum time control of a double integrator with inequality constraints on control (acceleration) and state (velocity and hence position) [30]. For more details about this optimal control theory result, readers are referred to Appendix A. Using these closed form expressions as constraints on robot path segment traversal times, the authors construct a Mixed Integer Nonlinear Program (MINLP) with additional constraints on robot vehicle kinematics, dynamics, and collision avoidance.

In [42], 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) [26] involving Laplacian matrices of graphs that represent target visibility and inter-robot communication connectivity. This work demonstrates the use of spectral graph theory and semi-definte programming techniques to solve the target tracking problem.

In [8], we consider a scenario with a group of car-like vehicles traversing known and fixed paths. These vehicles must begin their transit from a set starting point while making sure that a certain level of communication connectivity is maintained throughout the journey, and the communication graph is connected at all times. The problem is formulated and solved as a nonlinear program (NLP). We develop "Partition Elimination" constraints that assist in ensuring that the

communication network is fully connected (no network partitions). These constraints are enforced only when network partitions would otherwise occur, an approach which significantly reduces the problem size and the required computational effort. The vehicles must travel in such a way that the maximum transit time is minimized. The fixed paths of the vehicles are represented by spline curves. A comprehensive scalability test of our approach was performed and documented by testing scenarios with up to 50 vehicles.

As we stated, in its most general form, a MVMP problem is expressed by the Mathematical Program (3.1), and [8, 42, 122, 109] are excellent examples of the use of (3.1). In the following discussions, we highlight details about the formulations and results discussed in these four studies.

3.2 Decision Variables ζ and σ

Depending on the task to be performed and modeling approach adopted, the decision variables ζ and σ, for (3.1) are reference speed of each vehicle [122], segment traversal times and speeds at the start of each segment of a discretized path space [109], positions of the robot vehicles in continuous Euclidean space [42], or speed along the fixed paths represented by cubic spline path primitives [8]. For centralized formulations in [109, 42, 8], the values of all decision variables are determined simultaneously. For decentralized formulations presented in [122] and [9], each vehicle determines the values of its own speed in a sequential manner.

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

In the decentralized formulation of Schouwenaars et al. [122], $\zeta = \mathbf{u}_{\mathrm{k}}$, where $\mathbf{u}_{\mathrm{k}}$ is the reference speed of each vehicle at current time step k, and the objective function (3.2) for each vehicle $\Phi_a(\zeta)$ is a piecewise linear cost function that is comprised of the weighted ℓ_1-norm of the difference of the current state $\mathbf{x}_{\mathrm{k}}$ with the desired state $\mathbf{x}_{\mathrm{f}}$, minus the scalar product of the current velocity vector $\mathbf{s}_{\mathrm{k}}$ with the vector indicating the direction from initial position $\mathbf{p_0}$ to the final position $\mathbf{p_f}$. The

values of $\mathbf{x}_\mathrm{k}$ and $\mathbf{s}_\mathrm{k}$ depend on the decision variable $\zeta = \mathbf{u}_\mathrm{k}$. q and r are weights that are tuned appropriately. $\Phi_a(\zeta)$ provides a minimum time formulation.

$$\Phi_a(\zeta) = \sum_{k=1}^{T} [(q'|\mathbf{x}_\mathrm{k} - \mathbf{x}_\mathrm{f}|) - r(\mathbf{p}_\mathrm{f} - \mathbf{p}_0)'\mathbf{s}_\mathrm{k}], \tag{3.2}$$

In [109], the scenario completion time, which is the maximum time taken by the last arriving robot, is minimized. To maximize the tracking visibility, Derenick et al. in [42] minimize the negative of the second smallest eigenvalue of the visibility graph.

The centralized formulation in [8] is a minimum time formulation with $\zeta = \mathbf{s}_\mathrm{k}$, where $\mathbf{s}_\mathrm{k}$ is a vector of (continuous valued) speeds of all robots along their paths at discrete time step k. The formulation minimizes $\Phi_b(\zeta)$, where

$$\Phi_b(\zeta) = \mathbf{T}_\mathrm{max} + \gamma \sum_{i,k} \mathbf{d}^{\mathbf{i}_\mathrm{goal}(\mathbf{k})} \tag{3.3}$$

$\mathbf{d}^{\mathbf{i}}_\mathrm{goal}(\mathbf{k})$ is the distance from goal for robot $i = 1,\ldots,n$ at time step $k = 1,\ldots,T$. γ is a tunable weight with appropriate units. $\mathbf{T}_\mathrm{max}$ represents the traversal time of the last arriving robot and is defined using a discrete variable $A^i(k)$ for each robot $i = 1,\ldots,n$ at each time step $k = 1,\ldots,T$ as

$$A^i(k) = \begin{cases} 0, & \text{if } (\mathbf{d}^i_\mathrm{goal}(k) \neq 0) \\ 1, & \text{if } (\mathbf{d}^i_\mathrm{goal}(k) = 0) \end{cases} \tag{3.4}$$

$$\mathbf{T}_\mathrm{max} = \max_{i=1,\ldots,n} \left(\sum_{\mathrm{k}=0,\ldots,T} (1 - A^i(k)) \right) \tag{3.5}$$

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

To represent vehicle kinematics, Schouwenaars et al. in [122] use a discretized velocity control model, Peng and Akella use a double integrator model in [109], while Derenick et al. use a single order fully actuated model [42]. In [8], piecewise cubic spline functions with continuous first and second derivatives are used to represent the fixed paths. The spline

curve $ps^i(u^i(k))$ defines the position $(x^i(k), y^i(k))$ for each robot i at time step k. $u^i(k)$ is the arc-length traversed by the robot i along its spline path at time k. $u^i(k)$ depends on the speed $s^i(k)$ and time-step Δt (4.6).

$$\begin{aligned} (x^i(k), y^i(k)) &= ps^i(u^i(k)) \\ u^i(0) &= 0 \\ u^i(k) &= u^i(k-1) + s^i(k)\Delta t \end{aligned} \tag{3.6}$$

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

Schouwenaars et al., in [122], linearize the non-linear dynamic bounds by using N-sided polygonal approximations of a circle that represents dynamic bounds on speed. Readers are referred to Appendices C and D for more details. Non-linear bounds on speed and acceleration are used in [109] and [42] (second-order conic inequalities). In [8], due to the fact that the cubic splines representing the robot paths are twice continuously differentiable, the non-holonomic kinematic model used for two-wheeled differential drive robots with additional upper and lower bounds on speed and acceleration results in feasible motion at all times.

Both [109] and [122] are good examples of using Big-M constraints to express several *either–or* constraints.

3.6 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 [122], Schouwenaars et al. represent collision avoidance using loiter circles. For more details on loiter circles, readers are referred to Appendix E. In [42], Derenick et al. use conditions on the Euclidean Distance Matrix (EDM) [61] to model collision avoidance constraints, while in [8], a safety margin in terms of Euclidean distances is enforced on each vehicle at all discrete time steps.

Discretizing different aspects of the problem changes the nature of these constraints. In [109], by discretizing space instead of time, Peng and Akella express collision avoidance constraints by using segment traversal times. The authors note that if two robots i and j

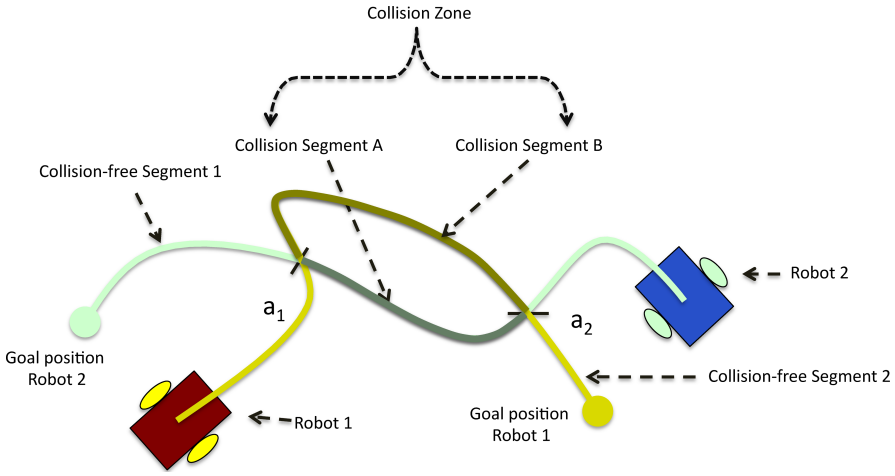


Fig. 3.1 Discretization of space: each robot's path is divided into one or more collision segments and collision-free segments. Any pair of points (one each from individual collision segments A and B) between and including points a_1 and a_2 form a collision pair. The individual collision segments A and B together form a collision zone.

could potentially collide if simultaneously traversing segments z_i and z_j respectively, then one of the following must hold to avoid collisions:

$$t^j_{z_j} \geq t^i_{z_i+1} \tag{3.7}$$

$$t^i_{z_i} \geq t^j_{z_j+1} \tag{3.8}$$

where t 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 are converted into (3.6).

$$\begin{gathered} t^j_{z_j} - t^i_{z_i+1} + M(1-\delta^{ij}_{z_i z_j}) \geq 0 \\ t^i_{z_i} - t^j_{z_j+1} + M\delta^{ij}_{z_i z_j} \geq 0 \\ \delta^{ij}_{z_i z_j} \in \{0,1\} \end{gathered} \tag{3.9}$$

$$\delta^{ij}_{z_i z_j} = \begin{cases} 1, & \text{if robot } i \text{ goes first along segment } z_i \\ 0, & \text{if robot } j \text{ goes first along segment } z_j, \end{cases} \tag{3.10}$$

where M is an arbitarily large number [19].

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

Communication requirements are mostly specified in terms of connectivity between the vehicles. While [122] assumes that there is a centralized communication hub that takes care of all communication needs, [42] and [8] explicitly deal with these communication requirements. In [42], Derenick et al. use constraints on the Laplacian matrix of 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 an n-dimensional square matrix whose entries are denoted by

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

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 an $n \times (n-1)$ matrix [77]. For more details, readers are referred to Appendix B.

In [8], we use the Friis's free-space communication model described in (2.11) to capture the propagation and power loss in the radio communication channels [54].

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

In [42], 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. 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

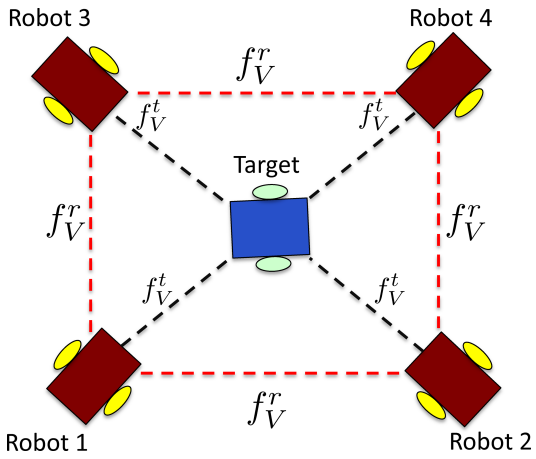


Fig. 3.2 A visibility graph with four robots tracking a target. f_V^r is the inter-robot link weight and f_V^t is the robot–target link weight.

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 is [77]. For more details, readers are referred to Appendix B.

4

Formulating and Solving Multi-Vehicle Motion Planning Problems

We are now ready to formulate, solve, and experimentally validate a Multi-Vehicle Motion Planning problem. In order to illustrate this, we first focus on the problem of Multi-Vehicle Path Coordination (MVPC) under communication connectivity constraints; a problem that features path-constrained vehicles that begin their transit from a fixed starting point and move toward a goal point along fixed paths so as to avoid collisions with other robots and static obstacles. The MP framework presented here allows to solve for time optimal velocity profiles for such robots in the presence of constraints on kinematics, dynamics, collision avoidance, and inter-robot communication connectivity. We formulate the problem as a distributed Receding Horizon Mixed Integer Nonlinear Programming (RH-MINLP). These sections are mostly drawn from our previous work [9]. We then synthesize experimental approaches that have been documented in the literature. Specifically, we describe experiments involving up to five ground robots operating in a reasonably complex workspace that were reported in our previous work [13], and experiments involving trajectory generation of four heterogeneous quadrotors operating in an indoor environment using Mixed Integer Quadratic Programming (MIQP) as documented in [95]. Results demonstrate the effect of communication connectivity requirements on

robot velocity profiles and the effect of sensing and actuation noise on the trajectory execution of the robots.

4.1 Formulating and Solving the Multi-Vehicle Path Coordination Problem

As defined in Section 2.1, the problem of Multi-Vehicle Path Coordination (MVPC) is formally defined as follows: *Given a group of vehicle robots that have fixed and known paths connecting an initial and a goal location, generate time-optimal velocity profiles that satisfy kinematic, dynamic, collision avoidance, and communication connectivity constraints.* The MVPC problem is an important one since more often than not, one does not get the liberty of planning an arbitrary path around sparse obstacles, and rather, must follow a prescribed route. This is especially true in situations where multiple driverless car-like vehicles operate in an urban environment. Most closely related to the work presented here is [122], where the authors present a decentralized receding horizon formulation for multiple aircraft path planning using MILP to generate provably safe trajectories. Similar to [122], the solution algorithm presented here features a sequential decision ordering mechanism. To the best of our knowledge, [13] presented the first experimental results for MVPC under communication connectivity constraints using RH-MINLP.

The three main elements of the problem — robot paths, inter-robot communication, and receding horizon planning — are presented in the following sections.

4.1.1 Robot Motion and Fixed Paths

Consider a group of N two-wheeled differential drive mobile robots shown in Figure 4.3. The robots move in a global (X,Y) Cartesian coordinate plane and are represented by the following kinematic model with associated non-holonomic constraints (that disallow the robot from sliding sideways).

$$\dot{x} = s\cos(\theta); \quad \dot{y} = s\sin(\theta); \quad \dot{\theta} = \omega \tag{4.1}$$

$$\dot{x}\sin(\theta) - \dot{y}\cos(\theta) = 0. \tag{4.2}$$

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.

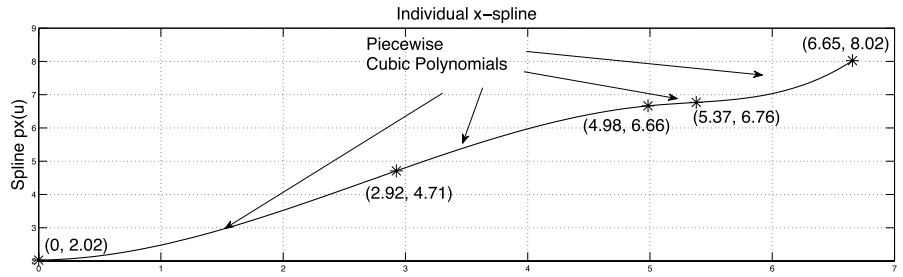
Each robot $i = 1,\ldots,N$ has a given start (origin) point o^i and a given end (goal) point e^i. O is the set of all start (origin) points. $o^i \in O, \forall i = 1,\ldots,N$. E is the set of all end points. $e^i \in E, \forall i = 1,\ldots,N$. The Euclidean distance between two robots i and j is denoted by d^{ij}. The robots are required to maintain a minimum safe distance d_{safe} from each other in order to avoid collisions. At any given discrete time step, the distance between the current location and the goal point for robot i is given by d^i_{goal}. s^i and ω^i denote the speed and angular velocity, respectively, of robot i along its path at a given time.

Each robot i follows a fixed path represented by a two-dimensional piecewise cubic spline curve of length U^i, which is obtained by combining two one-dimensional piecewise cubic splines $x(u)$ and $y(u)$, where the parameter u is arc length along the curve. Let $\kappa(u)$ be the curvature along the spline curve.

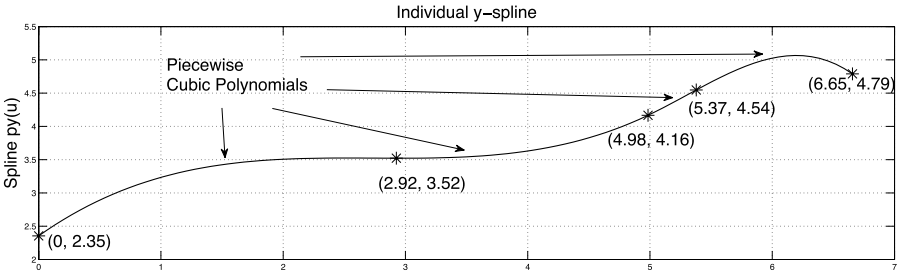
For each robot i,

$$\omega^i(u) = s^i(u)\kappa^i(u) \tag{4.3}$$

These piecewise cubic splines have continuous first derivatives (slope) and second derivatives (curvature) along the curve. This property makes the path kinematically feasible. Furthermore, upper and lower bounds on the speed, acceleration, and angular speed (turning rate) are enforced, thereby taking the robot dynamics into account. The paths represented by the two-dimensional piecewise cubic splines, along with the constraints on speed, accelerations and turn rates, result in a kinodynamically feasible trajectory. For a detailed discussion on spline curve design and analysis, see [97] and its references. Figures 4.1 and 4.2 show how the spline curve paths are constructed. Other path primitives that result in twice continuously differentiable functions in our constraints such as quintic curves, polar splines, and cubic spirals are easily accommodated using this framework.



(a) Arc Length u



(b) Arc Length u

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

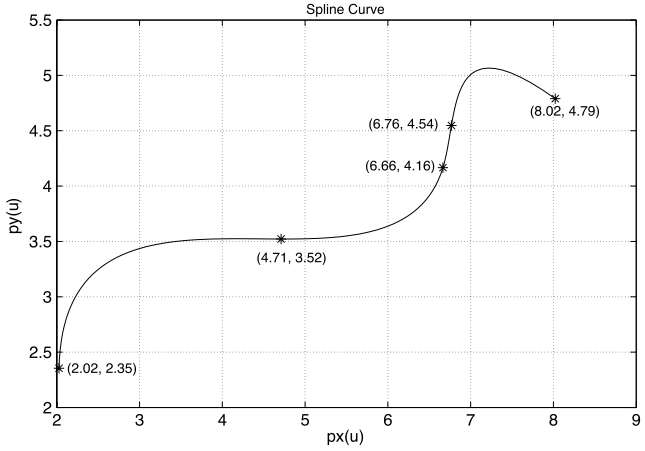


Fig. 4.2 The spline curve is obtained by combining $px(u)$ and $py(u)$.

4.1.2 Communication Model

Each robot is equipped with a wireless transceiver node. Consider two robots that try to communicate with each other at a given point of time. The Euclidean distance between them is denoted by d. The signal transmission power of the wireless node placed on the transmitter robot is denoted by P_{tr}. The received signal power of the wireless node placed on the receiver robot is denoted by P_{r}. The power experienced by the receiver robot node is calculated using Friis's equation provided in (2.11).

The SNR experienced by the receiver robot is calculated using the relationship $\mathrm{SNR} = P_{\mathrm{r}}/\sigma$ to determine whether the robots are in communication range of each other. 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 communication range of each other. For a known P_{tr} and α, the condition $\mathrm{SNR} \geq \eta_{\mathrm{c}}$ is modified appropriately as $d \leq \eta_{\mathrm{d}}$, where η_{d} defines the maximum communication range beyond which path-loss results in loss of communication.

4.1.3 Receding Horizon

The parameter t represents 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 until the mission is over. However, if required, it can still communicate with other robots.

Each robot plans its own trajectory by taking into account the plans of all other robots at each discrete time step. For a given time step t, each robot determines its speed for the next T_{hor} time steps starting at time t, i.e., $s^i(t)$, …, $s^i(t + T_{\mathrm{hor}})$ and implements the first speed $s^i(t + 1)$ 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: $\mathcal{P}^i(t) = [\mathbf{p}^i(t) \cdots \mathbf{p}^i(t + T_{\text{hor}})]$, 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 $s^i(t)$.

4.1.4 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 $\text{ord}(i)$, $i \in \{1, \ldots, N\}$.

4.2 Optimization Model Formulation

Each robot i solves the optimization problem $\mathcal{O}^i(t)$ indicated by (4.4)–(4.16) in the order $\text{ord}(i)$ that it has been assigned.

4.2.1 Decision Variables

In (4.4)–(4.16), the main decision variables are the speeds, $s^i(t), \ldots, s^i(t + T_{\text{hor}})$, for robot i at time t. The values of the remaining variables are dependent on the speeds.

4.2.2 Objective Function

Equation (4.4) 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 (4.12) defines the distance to goal $d^i_{\text{goal}}(k)$ for each robot $i = 1, \ldots, N$ at time-step k, $\forall k \in \{t, \ldots, t + T_{\text{hor}}\}$ as the difference between its 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).

4.2.3 Path (Kinematic) Constraints

Constraints (4.5)–(4.9) define the path of each robot. The constraints (4.5), (4.6), and (4.7) form the boundary conditions. Constraint (4.5)

indicates that each robot i has to start at a designated start point o^i. Constraint (4.6) initializes the arc length travelled u to zero value. Constraint (4.7) provides the upper bound on the arc length travelled. Constraint (4.8) increments the arc length at each time step based on the speed of the robot ($\Delta t = 1$). Constraint (4.9) 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, $\forall\, k \in \{t, \ldots, t + T_{\text{hor}}\}$ after travelling an arc length of $u^i(k)$ along the piecewise cubic spline curves. It should be noted that the constraint (4.9) is a non-convex nonlinear equality constraint.

$$\begin{aligned}
\text{minimize} \quad & \sum_{k=t}^{t+T_{\text{hor}}} d^i_{\text{goal}}(k) && (4.4)\\
\text{subject to} \quad & \forall\, j \in \{1, \ldots, N\}, j \neq i \\
& \forall\, k \in \{t, \ldots, t + T_{\text{hor}}\} \\
& (x^i(0), y^i(0)) = o^i && (4.5)\\
& u^i(0) = 0 && (4.6)\\
& u^i(k) \leq U^i && (4.7)\\
& u^i(k) = u^i(k-1) + s^i(k)\Delta t && (4.8)\\
& (x^i(k), y^i(k)) = ps^i(u^i(k)) && (4.9)\\
& s_{\min} \leq s^i(k) \leq s_{\max} && (4.10)\\
& a_{\min} \leq a^i(k) \leq a_{\max} && (4.11)\\
& d^i_{\text{goal}}(k) = U^i - u^i(k) && (4.12)\\
& d^{ij}(k) \geq d_{\text{safe}} && (4.13)\\
& d^{ij}(k) \leq M(1 - C^{ij}(k)) + \eta_{\text{d}} && (4.14)\\
& \sum_{j:j\neq i} C^{ij}(k) \geq n_{\text{conn}} && (4.15)\\
& C^{ij}(k) \in \{0, 1\} && (4.16)
\end{aligned}$$

4.2.4 Speed and Acceleration (Dynamic) Constraints

Constraints (4.10)–(4.11) are dynamic constraints and ensure that the speed $s^i(k)$ (and hence, angular velocity) and the acceleration $a^i(k)$ for each robot $i = 1, \ldots, N$ at each time-step k, $\forall k \in \{t, \ldots, t + T_{\text{hor}}\}$ 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 $\kappa(u)$ 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 is determined by (4.3).

4.2.5 Collision Avoidance Constraint

The constraint (4.13) ensures that there is a sufficiently large distance d_{safe} between each pair of robots to avoid a collision at all times. In addition to constraint (4.9), constraint (4.13) is another reason why $\mathcal{O}^i(t)$ has a relaxation that is a non-convex nonlinear programming problem.

4.2.6 Communication Connectivity Constraints

Constraints (4.15) and (4.16) state that vehicle i should be in a communication range of at least n_{conn} vehicles. This means that, for at least n_{conn} values of $j = 1, \ldots, N$, $j \neq i$, the condition $d^{ij} \leq \eta_{\text{d}}$ should be satisfied. The remaining vehicles may or may not be in the communication range of i. In order to express this requirement, we introduce a constant M and formulate constraint (4.14). In the case $d^{ij}(t) \leq \eta_{\text{d}}$, $C^{ij}(t)$ can assume a value of either 0 or 1. But the constraint (4.16) 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 $C^{ij} = 1$. If $d^{ij} > \eta_{\text{d}}$, then $C^{ij}(k)$ will have to equal 0 in order to satisfy the constraint (4.14). Constraint (4.14) is an example of a big-M constraint [19].

In the numerical implementation, an equivalent form of constraint (4.14) as

$$\frac{(d^{ij}(k))^2}{(M(1-C^{ij}(k))+\eta_{\mathrm{d}})} \leq (M(1-C^{ij}(k))+\eta_{\mathrm{d}}) \tag{4.17}$$

is used, 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.

4.2.7 RH-MINLP 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 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, $\forall j \in \{1,\ldots,N\},\ j \neq i$ whose $\mathrm{ord}(j) < \mathrm{ord}(i)$ — these robots have already calculated their new plans, and
  - 1.2 Plans $\mathcal{P}^\zeta(t)$ for robots $\zeta, \forall \zeta \in \{1,\ldots,N\}, \zeta \neq i$ whose $\mathrm{ord}(i) < \mathrm{ord}(\zeta)$ — 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 $T_{\text{hor}} - 1$ time steps, i.e., the new plan

$$\mathcal{P}^i(t+1) = \mathcal{P}^i(t) \backslash \mathbf{p}^i(t) \tag{4.18}$$

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

4.3 Experimental Approach

We have discussed the formulation and solution process of MVMP problems in the previous sections. From an experimental point of view, several commonalities have been observed in the technical approach reported in the literature. Figure 4.3 describes a general technical approach that has been utilized in two recent works [13, 95]. In [13],

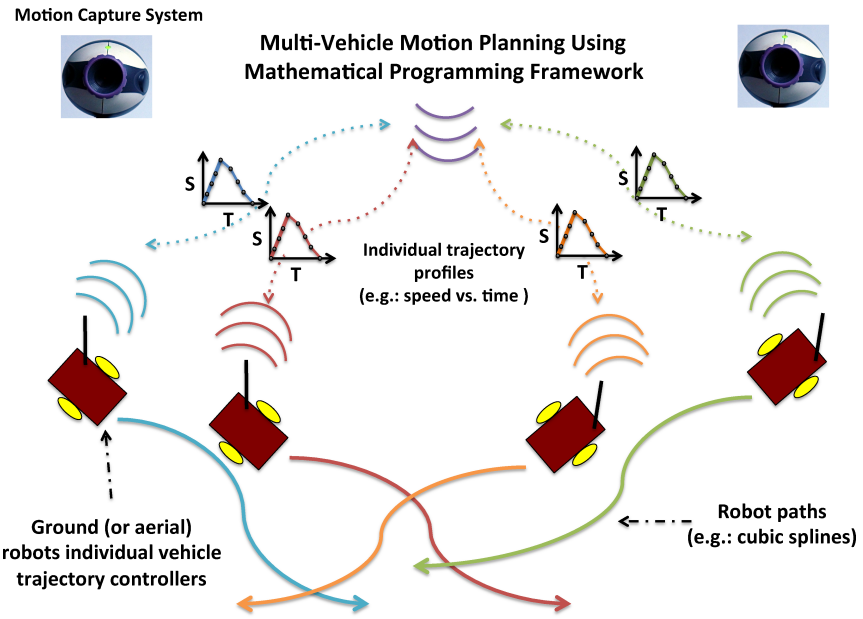


Fig. 4.3 A general experimental approach for implementing MP based MVMP under communication connectivity constraints.

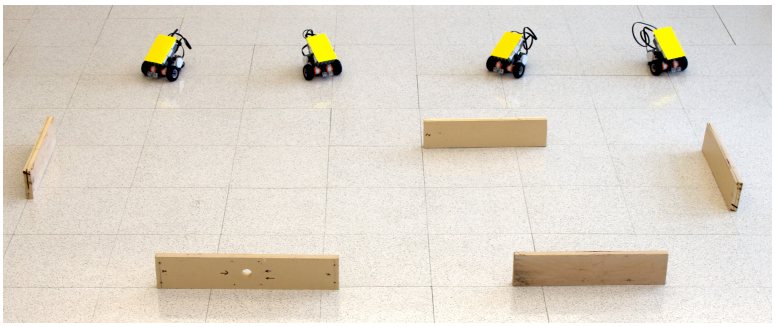


Fig. 4.4 Experimental setup similar to the ones utilized in [13].

a group of two-wheeled differential drive mSRV-1 robot vehicles travel along fixed and known piecewise cubic spline paths in a $4.5 \times 4.5\,\mathrm{m}$ workspace while maintaining communication connectivity with n_{conn} neighboring vehicles. 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. Localization for the individual robots was provided by a network of overhead cameras. A low-level trajectory-following PID controller was provided for each robot to implement trajectory commands. The distributed decision-making and computations were simulated on a central computer using MATLAB, which interfaced with the MINLP solver MILANO [21, 22]. This computer communicated the optimal speed profile information to each robot via a 802.11b wireless communication link. In [95], the authors utilized a micro-quadrotor testbed. The experimental setup used a motion capture system to estimate the position and velocity of the quadrotors. An inertial measurement unit (IMU) was utilized to estimate their orientation and angular velocity. CPLEX was used to solve all MIQP problems in real-time and communicate trajectory information to all robots.

4.4 Experimental Results and Insights

Using the experimental approach described earlier, both [95] and [13] performed indoor experiments to validate MP-based motion planning

approaches. In [13], two different types of experiments were conducted. The first involved pre-solving the MVPC before communicating the speed profiles to the robots. The second set of experiments involved real-time speed profile generation by utilizing real-time location feedback to solve the RH-MINLP. Similar to the second set of experiments in [13], [95] performed experiments with quadrotors by solving the MIQP formulation in real-time while incorporating feedback from the motion capture system and communicating the trajectory commands to the controllers onboard the quadrotors.

The results of the MVPC experiments conducted in [13] reveal a strong interplay between communication connectivity requirements and the speed profiles of the robots. It was found that while the optimization allowed for more stringent communication connectivity requirements without significantly degrading the scenario completion time, the speed profiles of the individual robots were affected. In [13], it was noted that typically, the robots whose times of arrival at their respective destinations are less than the scenario completion time change their speed profiles to comply with the new communication constraint. It was also observed that even with more stringent communication connectivity requirements, there was no appreciable increase in the scenario completion times. Again, this was attributed to the fact that the faster robots slow down to accommodate the more stringent communication connectivity requirements.

In [95], enforcing constraints on positions, velocities, accelerations, jerks, and quadrotor inputs allowed the authors to perform experiments with two quadrotors passing through a gap in 3-D, three quadrotors moving in a plane with obstacles, and a heterogeneous group of four quadrotors flying in reconfigurable formations.

Post-processing the overhead localization data allows to quantify the effects of sensing and actuation noise on the performance of the robots by calculating the root-mean squared error (RMSE) between the desired and actual robot trajectories. The authors in [13] noted that the RMSE becomes particularly pronounced when the curvature of the fixed paths changes rapidly between successive discrete time steps. On the other hand, the authors in [95] noted that with larger acceleration, jerk, and snap in the desired trajectory, the RMSE value

increases. However, the performance degradation due to this increase is graceful.

Sequential receding horizon planning does come with its own limitations. In [13], the authors note that it was found the decision order $\text{ord}(i)$ of the robots $i = 1, \ldots, n$ can qualitatively affect the solution of each robot depending on the geometry of the paths and that certain robots' decisions may render the coordination problem difficult to solve for other robots. It was found that reassigning a different decision order $\text{ord}(i)$ of the robots $i = 1, \ldots, n$ helped improve overall solutions.

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 planning algorithms must provide some mechanism to ensure that the robots are in a safe state. In [122], the authors ensure that in case of situations when there is no solution available, the robots enter a loiter circle state.

5

Observations and Design Considerations

In this monograph, we have presented a unified and general mathematical programming framework for modeling and solving multi-vehicle motion planning problems. We have given a thorough review of the literature and have highlighted the use of this framework in trajectory planning for autonomous aerial vehicles, path coordination of ground robots under communication constraints, and collaborative target tracking. A complete demonstration of a distributed path coordination problem and its solution has been provided.

Mathematical programming has been shown to be a flexible and powerful framework for modeling in these scenarios. Existing modeling environments and solvers are key to quickly developing and experimenting with models. For the future development of research in MVMP, there are many advantages of the framework to consider and several important challenges that pose interesting research problems in the years to come.

5.1 Advantages accorded by MP

5.1.1 Concepts/Results from Other Disciplines

The framework is general enough to incorporate concepts and results from other disciplines. In [122], the mathematical program is embedded in a decentralized decision-making algorithm. In [109], the authors make use of results from optimal control theory to get bounds on the decision variables, while in [42], the authors use results from spectral graph theory to express visibility and communication constraints. In [8], we were able to reduce the problem size and computational effort by using an algorithmic approach that introduced constraints only when required. In recent work [17], 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 [57]. The output of the first stage acts as an initial guess to second-stage NLP formulations.

5.1.2 Discretization of Space and/or Time

The framework is general enough to allow discretization of space and/or time. Studies [42, 122], and [8] are examples of continuous space and discrete time formulations. In [109], 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 [30].

5.1.3 Path Primitives

The framework is general enough to allow any path primitives such as straight lines and circular arcs [109], and two-dimensional cubic splines [8] among others to be used.

5.1.4 Centralized/Decentralized Decision Making

The framework allows for centralized decision making schemes as witnessed in case of [42, 109], and [8] or decentralized decision making schemes, wherein multiple decision makers independently solve their optimization problems [9, 122].

5.1.5 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 [122, 128].

5.1.6 Handling Non-Convexity

When the MVMP problem has convex objective functions and convex feasible region, polynomial time algorithms are 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 is seen in [122], where the authors develop polygonal approximations of a circle to express dynamic constraints. The polygonal approximations of a circle are achieved in a more efficient way using the optimal linear formulations presented in [20]. On the other hand, the authors in [135] develop convex approximations of collision avoidance constraints by using the concept of optimal reciprocal collision avoidance (ORCA). Recent advances in MINLP solution techniques can be leveraged to solve MVMP variants without the need to linearize/approximate nonlinear, non-convex constraints [31]. Under certain conditions, lack of convexity due to discrete variables is handled using

branch and bound and cutting plan algorithms for MILP, and branch and bound and outer approximation algorithms for MINLP [31].

5.1.7 Modeling Environments, Solution Algorithms and Solvers

As mentioned earlier in Section 2.3.3, `AMPL` [52], `GAMS` [27], `MATLAB`, and `YALMIP` [91] can be used for rapid development of models for testing/simulation purposes. Section 2.3.2 lists a plethora of solution algorithms and solvers for these models.

Schouwenaars et al. in [122] implement their entire framework using `AMPL`. In particular, several large-scale NLP solution techniques handle non-convexities but do not provide guarantees of finding globally optimal solution. These include solvers like `LOQO` that uses interior-point methods [137], `KNITRO` that uses trust-region algorithms [32], and `SNOPT` that uses a quasi-Newton algorithm [60] among others. These solvers also incorporate mechanisms to detect infeasibilities and unboundedness in the problem. Solvers such as `CPLEX` [66] heavily preprocess the problem before proceeding to solving them thereby speeding up the solution process tremendously. `CPLEX` has been successfully used to solve practical MVMP problems in real-time [118, 120]. Furthermore, in recent work, highly efficient code has been written to implement these algorithms in real time [93, 141].

5.2 Lessons Learned and Future Directions

Mathematical Programming (MP) provides a powerful framework for solving communication-centric multi-vehicle motion planning problems (MVMP). In this monograph, we introduced the readers to the foundations and trends in MP-based MVMP. Specifically, we discussed the building blocks, general framework, experimental implementation, and benefits and limitations of MP-based MVMP. We emphasized the fact that this framework lends itself to numerical solutions using readily available commercial and/or open-source solvers, and in many cases, via efficient polynomial time algorithms.

We reviewed several publications that use MP to solve variants of the general MVMP problem. We presented a general MP-based

framework that accommodates all the objectives and constraints of an MVMP problem, and we focused on four representative efforts that serve as examples of using this framework.

As an example to demonstrate real-world experimental studies, a decentralized RH-MINLP formulation for solving multi-vehicle path coordination problems under communication constraints was presented. Experimental approaches used in the literature to implement the MP frameworks were discussed. These results provide insights into the technical challenges associated with the implementation of MP-based motion planning.

While these results are encouraging, there are several challenges associated with the practical implementation of an MP-based motion planning framework that need to be addressed. These are briefly listed here:

- In addition to the kinodynamic, collision avoidance, and communication connectivity constraints, it is important to account for the constraints imposed from noise in the sensors, and the errors of the actuators in implementing the optimal commands perfectly. It became clear in [95] and [13] that while the RMSE values reflect the performance of lower-level robot controllers and allow for better gain selection, ultimately the control issues need to be incorporated in the MP formulations.
- Distributed decision-making for multi-vehicle systems is an active area of research [139]. The sequential decision-making algorithm demonstrated in this monograph provides a starting point to design more sophisticated distributed decision-making algorithms that are executed in conjunction with MP.
- We described several advantageous properties of the MP, including the ability to incorporate various path primitives, discretization of space and/or time, centralized/decentralized decision making, and receding horizon/model predictive control in a vast array of real-world settings. We noted that while MP-based frameworks prove to be effective for a small

number of robots, in order for them to be used with a large number of robots, the computational bottlenecks resulting from non-convexities and/or the size of the problem should be handled effectively. Efficient linear approximations for nonlinear functions help address this challenge. For example, the polygonal approximations of a circle provided in [122] are achieved in a more efficient way using the optimal linear formulations presented in [20]. In recent work [93], highly efficient code has been written to implement convex optimization solution algorithms in real time. The effectiveness of the algorithms presented in [93] in solving MVMP problems remains to be seen.

- The ultimate test of the proposed framework will be its application to real-world systems. This will require a collaborative effort between researchers and engineers belonging to both the MP and MVMP communities. In closing, as MP solution algorithms and numerical solvers continue to improve, we anticipate that this framework will be applied to a greater number of MVMP problems, and that the discussion presented in this monograph may serve as a guide for future MVMP research.

A

Time-Optimal Control of a Double Integrator

A minimum (maximum) time optimal control for a double integrator is described as follows:

$$\text{Minimize or Maximize } \Delta T = \int_0^{\Delta T} 1 \, \mathrm{d}t \tag{A.1}$$

subject to

$$\begin{pmatrix} \dot{x} \\ \dot{s} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ s \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} a(t) \tag{A.2}$$

$$x(0) = -S \quad x(\Delta T) = 0 \tag{A.3}$$

$$s(0) = s_{\text{start}} \quad s(\Delta T) = s_{\text{end}} \tag{A.4}$$

$$0 \leq s \leq s_{\max} \quad -a_{\max} \leq a \leq a_{\max} \tag{A.5}$$

x, s, and a are position, speed, and, acceleration of the double integrator system, respectively. The solution to the above problem is obtained by solving a standard two point boundary value problem (TPBVP) and turns out to be bang-bang or bang-off-bang in nature [30, 78].

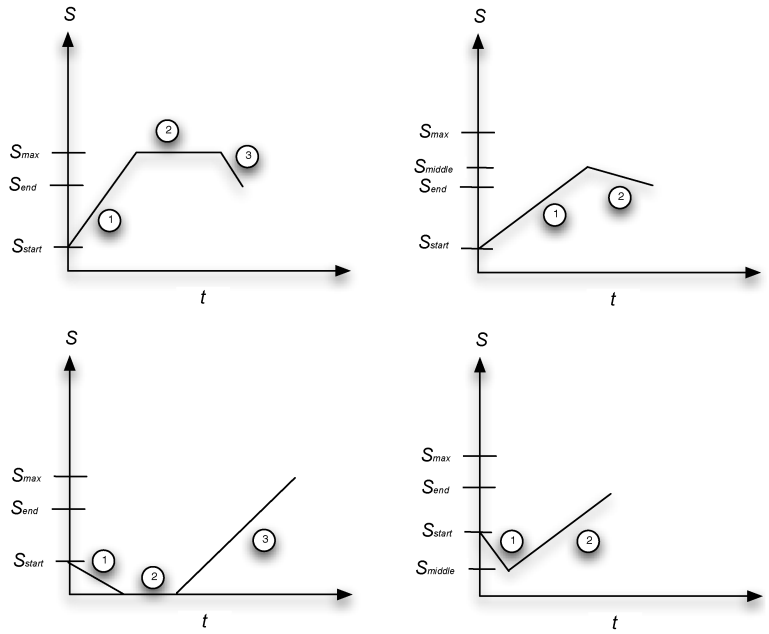


Fig. A.1 S is long enough to allow robot to reach $s_{\max}$ (top left), to reach zero speed (bottom left). When S is not long enough to allow the robot to reach max (top right) and zero (bottom right) speed, the robot moves at s_{middle} speed.

Accordingly there are four cases possible depending on whether the segment length S is long enough to allow the robot to reach maximum (and zero) velocity.

(1) **Minimum time with S being long enough to allow the robot to reach $s_{\max}$:** this scenario is depicted in the top left subplot of Figure A.1 where the robot ramps up to max-speed (segment 1), stays at max speed (segment 2) and then decelerates at max-speed (segment 3). Given a path of length S with start speed denoted by s_{start} and speed at the end of the segment denoted by s_{end}, the following relationships hold true:

$$S = s_{\text{start}} t \pm \frac{1}{2} a_{\max} t^2 \tag{A.6}$$

$$s_{\text{end}} = s_{\text{start}} \pm a_{\max} t \tag{A.7}$$

For segment 1, whose length is assumed to be S_1, to ascertain max-speed, Equations (A.7) and (A.6) are utilized as follows:

$$s_{\max} = s_{\text{start}} + a_{\max} t \Rightarrow t = \frac{s_{\max} - s_{\text{start}}}{a_{\max}} \tag{A.8}$$

$$S_1 \geq s_{\text{start}} \left(\frac{s_{\max} - s_{\text{start}}}{a_{\max}} \right) + \frac{1}{2} a_{\max} \left(\frac{s_{\max} - s_{\text{start}}}{a_{\max}} \right)^2 \tag{A.9}$$

$$S_1 \geq \frac{1}{2} \left(\frac{s_{\max}^2 - s_{\text{start}}^2}{a_{\max}} \right) \tag{A.10}$$

Also, traversal time is calculated as follows:

$$\Delta t_1 = \frac{s_{\max} - s_{\text{start}}}{a_{\max}} \tag{A.11}$$

For segment 3, whose length is assumed to be S_3, using Equation (A.7) and then (A.6), the following relationships hold true:

$$s_{\text{end}} = s_{\max} - a_{\max} t \Rightarrow t = \frac{s_{\max} - s_{\text{end}}}{a_{\max}} \tag{A.12}$$

$$S_3 \geq s_{\max} \left(\frac{s_{\max} - s_{\text{end}}}{a_{\max}} \right) - \frac{1}{2} a_{\max} \left(\frac{s_{\max} - s_{\text{end}}}{a_{\max}} \right)^2 \tag{A.13}$$

$$S_3 \geq \frac{1}{2} \left(\frac{s_{\max}^2 - s_{\text{end}}^2}{a_{\max}} \right) \tag{A.14}$$

Also, traversal time

$$\Delta t_3 = \frac{s_{\max} - s_{\text{end}}}{a_{\max}} \tag{A.15}$$

Using (A.10) and (A.14), we see that for this scenario to be feasible, the total path length, $S \geq S_1 + S_3$, i.e.,

$$S \geq \frac{1}{2} \left(\frac{s_{\max}^2 - s_{\text{start}}^2}{a_{\max}} \right) + \frac{1}{2} \left(\frac{s_{\max}^2 - s_{\text{end}}^2}{a_{\max}} \right) \tag{A.16}$$

Based on the above results, the traversal time for segment 2 is described as

$$\Delta t_2 = \frac{S}{s_{\max}} - \frac{1}{2}\left(\frac{s_{\max}^2 - s_{\text{start}}^2}{a_{\max} s_{\max}}\right) - \frac{1}{2}\left(\frac{s_{\max}^2 - s_{\text{end}}^2}{a_{\max} s_{\max}}\right) \tag{A.17}$$

Thus the total traversal time for this segment is obtained from Equations (A.11), (A.17), and (A.15)

$$\Delta t = \Delta t_1 + \Delta t_2 + \Delta t_3 \tag{A.18}$$

$$\Delta t = \frac{s_{\max} - s_{\text{start}}}{a_{\max}} + \frac{S}{s_{\max}} - \frac{1}{2}\left(\frac{s_{\max}^2 - s_{\text{start}}^2}{a_{\max} s_{\max}}\right) - \frac{1}{2}\left(\frac{s_{\max}^2 - s_{\text{end}}^2}{a_{\max} s_{\max}}\right) + \frac{s_{\max} - s_{\text{end}}}{a_{\max}} \tag{A.19}$$

(2) **Minimum time with S not allowing the robot to reach $s_{\max}$:** This scenario is depicted in top right subplot of Figure A.1. Assuming that the robot reaches $s_{\text{middle}} < s_{\max}$,

$$\Delta t = \Delta t_1 + \Delta t_2 = \frac{s_{\text{middle}} - s_{\text{start}}}{a_{\max}} + \frac{s_{\text{middle}} - s_{\text{end}}}{a_{\max}} \tag{A.20}$$

where, s_{middle} is obtained as follows:
For segment 1, whose length is assumed to be S_1,

$$s_{\text{middle}} = s_{\text{start}} + a_{\max} t \Rightarrow t = \frac{s_{\text{middle}} - s_{\text{start}}}{a_{\max}} \tag{A.21}$$

$$S_1 \geq s_{\text{start}}\left(\frac{s_{\text{middle}} - s_{\text{start}}}{a_{\max}}\right) + \frac{1}{2} a_{\max}\left(\frac{s_{\text{middle}} - s_{\text{start}}}{a_{\max}}\right)^2 \tag{A.22}$$

$$S_1 \geq \frac{1}{2}\left(\frac{s_{\text{middle}}^2 - s_{\text{start}}^2}{a_{\max}}\right) \tag{A.23}$$

For segment 2, whose length is assumed to be S_2

$$s_{\text{end}} = s_{\text{middle}} - a_{\max} t \Rightarrow t = \frac{s_{\text{middle}} - s_{\text{end}}}{a_{\max}} \tag{A.24}$$

$$S_2 \geq s_{\text{middle}} \left(\frac{s_{\text{middle}} - s_{\text{end}}}{a_{\max}} \right) - \frac{1}{2} a_{\max} \left(\frac{s_{\text{middle}} - s_{\text{end}}}{a_{\max}} \right)^2 \tag{A.25}$$

$$S_2 \geq \frac{1}{2} \left(\frac{s_{\text{middle}}^2 - s_{\text{end}}^2}{a_{\max}} \right) \tag{A.26}$$

$$S = S_1 + S_2 \Rightarrow s_{\text{middle}} = \frac{1}{2} \sqrt{2s_{\text{start}}^2 + 2s_{\text{end}}^2 + 4Sa_{\max}} \tag{A.27}$$

$$S = S_1 + S_2 \geq \frac{1}{2} \left(\frac{s_{\text{middle}}^2 - s_{\text{start}}^2}{a_{\max}} \right) + \frac{1}{2} \left(\frac{s_{\text{middle}}^2 - s_{\text{end}}^2}{a_{\max}} \right) \tag{A.28}$$

(3) **Maximum time with S being long enough to allow robot to reach zero velocity:** In this case (bottom left subplot of Figure A.1), once the robot reaches zero velocity, it can stay there for an infinite amount time before accelerating to its final destination, in this case,

$$\Delta t = \infty \tag{A.29}$$

(4) **Maximum time with S not allowing the robot to reach zero velocity:** This scenario is depicted in the bottom right subplot of Figure A.1. Assuming that the robot reaches $s_{\text{middle}} < s_{\text{start}} < s_{\max}$

$$\Delta t = \Delta t_1 + \Delta t_2 = \frac{s_{\text{start}} - s_{\text{middle}}}{a_{\max}} + \frac{s_{\text{end}} - s_{\text{middle}}}{a_{\max}} \tag{A.30}$$

where, s_{middle} is obtained as follows
For segment 1, whose length is assumed to be S_1,

$$s_{\text{middle}} = s_{\text{start}} - a_{\max} t \Rightarrow t = \frac{s_{\text{start}} - s_{\text{middle}}}{a_{\max}} \tag{A.31}$$

$$S_1 \geq s_{\text{start}} \left(\frac{s_{\text{start}} - s_{\text{middle}}}{a_{\max}} \right) - \frac{1}{2} a_{\max} \left(\frac{s_{\text{start}} - s_{\text{middle}}}{a_{\max}} \right)^2 \tag{A.32}$$

$$S_1 \geq \frac{1}{2} \left(\frac{s_{\text{start}}^2 - s_{\text{middle}}^2}{a_{\max}} \right) \tag{A.33}$$

For segment 2, whose length is assumed to be S_2,

$$s_{\text{end}} = s_{\text{middle}} + a_{\max} t \Rightarrow t = \frac{s_{\text{end}} - s_{\text{middle}}}{a_{\max}} \tag{A.34}$$

$$S_2 \geq s_{\text{middle}} \left(\frac{s_{\text{end}} - s_{\text{middle}}}{a_{\max}} \right) - \frac{1}{2} a_{\max} \left(\frac{s_{\text{end}} - s_{\text{middle}}}{a_{\max}} \right)^2 \tag{A.35}$$

$$S_2 \geq \frac{1}{2} \left(\frac{s_{\text{end}}^2 - s_{\text{middle}}^2}{a_{\max}} \right) \tag{A.36}$$

$$S = S_1 + S_2 \Rightarrow s_{\text{middle}} = \frac{1}{2} \sqrt{2 s_{\text{start}}^2 + 2 s_{\text{end}}^2 - 4 S a_{\max}} \tag{A.37}$$

$$S = S_1 + S_2 \geq \frac{1}{2} \left(\frac{s_{\text{start}}^2 - s_{\text{middle}}^2}{a_{\max}} \right) + \frac{1}{2} \left(\frac{s_{\text{end}}^2 - s_{\text{middle}}^2}{a_{\max}} \right) \tag{A.38}$$

B

Spectral Graph Theory and Graph Laplacian

We provide proofs of propositions that were directly used in [42]. These proofs were provided in the [77], and allow for a graph-theoretical approach to maintaining visibility in MVMP.

Theorem B.1. Consider the m-dimensional subspace $\mathbf{P} \subseteq \mathbb{R}^{\mathbf{n}}$ spanned by the vectors $p_i \in \mathbb{R}^n$, $i = 1,\ldots,m$. Denote $P := [p_1,\ldots,p_m] \in \mathbb{R}^{n\times m}$. Then for $M \in \mathbf{S}^n$ one has

$$x^T M x \succ 0 \text{ for all nonzero } x \in \mathbf{P} \text{ if and only if } P^T M P \succ 0 \qquad \text{(B.1)}$$

Proof. Any non-zero element of $x \in \mathbf{P}$ is described as

$$x := \alpha_1 p_1 + \alpha_2 p_2 + \cdots + \alpha_m p_m$$

for some $\alpha_1,\ldots,\alpha_m \in \mathbf{R}$, not all zeros and, thus $x = Py$, where y := $[\alpha_1,\ldots,\alpha_m]^T$. Consequently, the first inequality in (B.1)

$$x^T M x = (Py)^T M (Py) = y^T P^T M P y \succ 0 \qquad \text{(B.2)}$$

all nonzero $y \in \mathbf{R}^m$. □

Theorem B.2. For a graph Laplacian matrix $L(G)$, the condition $\lambda_2 > 0$ is equivalent to the transformation $P^T L(G) P \succ 0$, where P is a $n \times (n-1)$ matrix whose columns form an orthonormal basis for a subspace that is orthogonal to the vector $\mathbf{1}$. $\mathbf{1}$ is an n-dimensional vector that corresponds to the zero (0) eigenvalue of $L(G)$. All entires of $\mathbf{1}$ are equal to one (1).

Proof. A graph Laplacian is known to be a positive semidefinite matrix, i.e.,

$$L(G) \succeq 0 \quad \text{and} \quad L(G)\mathbf{1} = 0.$$

The smallest eigenvalue of $L(G)$ is always zero and $\mathbf{rank}\ L(G) \leq n-1$.

It is obvious that the similarity transformation described above will not change the eigenspace of $L(G)$. And so for the second smallest eigenvalue of $L(G)$ to be positive,

$$\lambda_2 > 0 \Longleftrightarrow x^T L(G) x \succ 0, \text{ for all non-zero } x \in \mathbf{1}^{\perp},$$
$$\text{where } \mathbf{1}^{\perp} := \{x \in \mathbf{R}^n | \mathbf{1}^{\perp} x = 0\}.$$

□

C

Polygonal Approximations of a Circle

Schouwenaars et al. present several linearization techniques for handling nonlinear constraints. One such technique is the polygonal approximation of a circle [120].

Consider a hexagon circumscribing circle with radius $V_{\max}$. As shown in the Figure C.1, let $\mathbf{V_0} = [\dot{x}; \dot{y}]^T$ be a point on the line containing one side of this polygon. Consider a vector $\mathbf{V}$ orthogonal to this line.

It is apparent from the figure that the equation of this line is given by the following dot product

$$
\begin{aligned}
&(\mathbf{V_0} - \mathbf{V}) \cdot \mathbf{V} = 0 \\
&\quad \Leftrightarrow \begin{bmatrix} \dot{x} - V_{\max}\cos\theta \\ \dot{y} - V_{\max}\sin\theta \end{bmatrix} \cdot \begin{bmatrix} V_{\max}\cos\theta \\ V_{\max}\sin\theta \end{bmatrix} = 0 \\
&\quad \Leftrightarrow \dot{x}V_{\max}\cos^2\theta - V_{\max}^2\cos^2\theta + \dot{y}V_{\max}\sin\theta - V_{\max}^2\sin^2\theta = 0 \\
&\quad \Rightarrow \dot{x}V_{\max}\cos\theta + \dot{y}V_{\max}\sin\theta = V_{\max}^2\cos^2\theta + V_{\max}^2\sin^2\theta \\
&\quad \Rightarrow \dot{x}\cos\theta + \dot{y}\sin\theta = V_{\max}
\end{aligned}
\tag{C.1}
$$

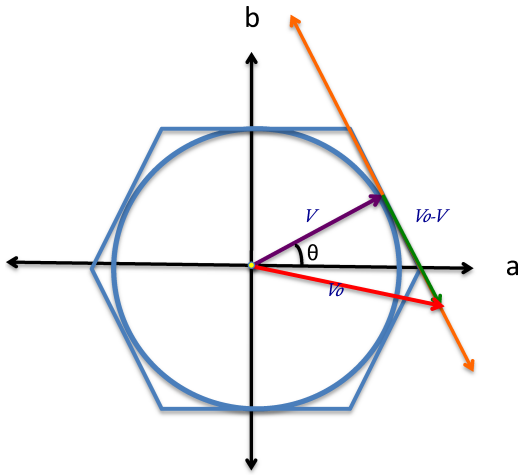


Fig. C.1 Approximation of a circle by a polygon (hexagon in this case).

Any point being approximated by this side of the N-sided polygon is represented by

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

This technique helps to convert a nonlinear constraint into a set of N linear constraints.

D

Approximation of An Ellipse by Two Intersecting Circles

The effective acceleration of a UAV flying in cruise mode is thought of as being bounded by an ellipse [120]. For a case where the maximum forward acceleration $a_{\text{fwd,max}}$ of the UAV is greater than the maximum lateral acceleration $a_{\text{lat,max}}$, the major axis of this ellipse is determined by $a_{\text{fwd,max}}$ while the minor axis of this ellipse is determined by $a_{\text{lat,max}}$.

In Figure E.1, the ellipse E indicates the bounds on the acceleration of the UAV traveling at speed v, acceleration a in the direction of the speed. $v^{\perp}$ is the component orthogonal to v.

This ellipse is approximated by the intersection of two identical circles, circle 1 and circle 2, both having radius β and centers located at a distance α from the center of the ellipse.

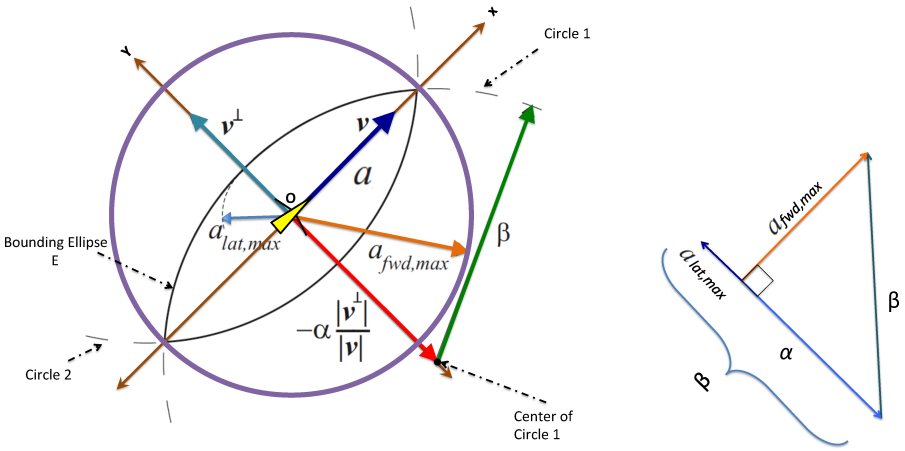


Fig. D.1 Approximation of an ellipse by two intersecting circles as depicted in [120].

α and β is ascertained as follows. From the triangle shown in Figure E.1, it is clear

$$\begin{aligned}
\beta &= \alpha + a_{\text{lat,max}} \\
\beta^2 &= \alpha^2 + a_{\text{fwd,max}}^2 \Rightarrow \beta^2 = (\beta - a_{\text{lat,max}})^2 + a_{\text{fwd,max}}^2 \\
&\Rightarrow \frac{(a_{\text{lat,max}}^2 + a_{\text{fwd,max}}^2)^2}{4a_{\text{lat,max}}^2} = \beta^2 \\
&\Rightarrow \frac{(a_{\text{fwd,max}}^2 - a_{\text{lat,max}}^2)^2}{4a_{\text{lat,max}}^2} + a_{\text{fwd,max}}^2 = \beta^2 \\
&\Rightarrow \alpha = \frac{(a_{\text{fwd,max}}^2 - a_{\text{lat,max}}^2)}{2a_{\text{lat,max}}} \quad \text{and} \\
\beta &= \sqrt{\frac{(a_{\text{fwd,max}}^2 - a_{\text{lat,max}}^2)^2}{4a_{\text{lat,max}}^2} + a_{\text{fwd,max}}^2}
\end{aligned}$$

E

Loiter Circles

Loiter circles were used in [120] to provide a guaranteed collision free trajectory planning.

$\mathbf{p}_T$ and $\mathbf{v_T}$ are the ingress position and velocity of the loiter circle at the end of the planning horizon.

$$\mathbf{p}_T = \begin{bmatrix} x(T) \\ y(T) \end{bmatrix} \quad \text{and} \quad \mathbf{v}_\mathbf{T}^\perp = \begin{bmatrix} -\dot{y}(T) \\ \dot{x}(T) \end{bmatrix}$$

From the geometry, it is clear that the right loiter circle is described by a set of points $\mathbf{p}_{\mathrm{R},\theta}$ where,

$$\begin{aligned} \mathbf{p}_{\mathrm{R},\theta} &= \mathbf{p}_\mathrm{R} + \mathbf{R}(\theta)(\alpha_\mathrm{c}\mathbf{v}_\mathbf{T}^\perp) \\ &= (\mathbf{p}_T - \alpha_\mathrm{c} v_T^\perp) + \mathbf{R}(\theta)(\alpha_\mathrm{c}\mathbf{v}_\mathbf{T}^\perp) \\ &= \mathbf{p}_T + \alpha_\mathrm{c}(\mathbf{R}(\theta) - \mathbf{I})\mathbf{v}_\mathbf{T}^\perp \\ &= \begin{bmatrix} x(T) \\ y(T) \end{bmatrix} + \alpha_\mathrm{c} \begin{bmatrix} \cos(\theta) - 1 & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) - 1 \end{bmatrix} \begin{bmatrix} -\dot{y}(T) \\ \dot{x}(T) \end{bmatrix} \end{aligned} \tag{E.1}$$

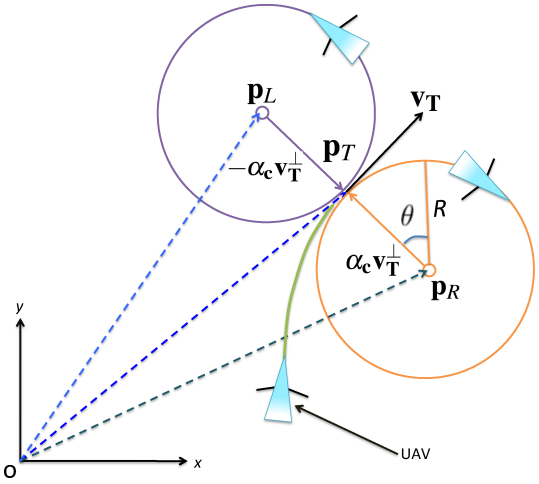


Fig. E.1 Geometry of loiter circles as depicted in [120].

and the left loiter circle is

$$\begin{aligned}
\mathbf{p}_{L,\theta} &= \mathbf{p}_L - \mathbf{R}(\theta)(\alpha_c \mathbf{v}_\mathbf{T}^\perp) \\
&= (\mathbf{p}_T + \alpha_c v_T^\perp) - \mathbf{R}(\theta)(\alpha_c \mathbf{v}_\mathbf{T}^\perp) \\
&= \mathbf{p}_T - \alpha_c(\mathbf{R}(\theta) - \mathbf{I})\mathbf{v}_\mathbf{T}^\perp \\
&= \begin{bmatrix} x(T) \\ y(T) \end{bmatrix} - \alpha_c \begin{bmatrix} \cos(\theta) - 1 & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) - 1 \end{bmatrix} \begin{bmatrix} -\dot{y}(T) \\ \dot{x}(T) \end{bmatrix}
\end{aligned} \tag{E.2}$$

$\mathbf{R}(\theta)$ is the standard rotation matrix. As seen here, the size of the loiter circles depends on the ingress velocity and is controlled effectively by the UAV.

Acknowledgments

The authors would like to thank Dr. M. Ani Hsieh, Dr. Matthew Mason, Dr. Jonathan How, Dr. Tom Schouwenaars, Dr. Kapil Dandekar, Dr. Steven Weber, Dr. Zvi. Shiller, Gabriel Ford, Jeff Wildman, Kenneth Mallory, James Milligan, and Scott Haney for their useful discussions.

Notations and Acronyms

LP: linear programming
MVMP: multi-vehicle motion planning
MVPC: multi-vehicle path coordination
MP: mathematical programming
MIP: mixed integer programming
MILP: mixed integer linear programming
MIQP: mixed integer quadratic programming
MINLP: mixed integer nonlinear programming
NLP: nonlinear programming
SDP: semi-definite programming
SOCP: second-order cone programming
SQP: sequential quadratic programming

References

[1] "BONMIN: Basic Open-source Nonlinear Mixed INteger programming," Available from: https://projects.coin-or.org/Bonmin/wiki.

[2] "GNU linear programming kit," Available from: http://www.gnu.org/software/glpk/.

[3] "GUROBI Optimizer," Available from: http://www.gurobi.com.

[4] "The MOSEK optimization software," Available from: http://www.mosek.com/.

[5] "Solving Constraint Integer Programs," Available from: http://www.scip.zib.de.

[6] "Xpress-mp and mosel," Available from: http://www.fico.com.

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

[8] P. Abichandani, H. Benson, and M. Kam, "Multi-vehicle path coordination in support of communication," in *Proceedings International Conference on Robotics and Automation*, Kobe, Japan, May 2009.

[9] 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.

[10] P. Abichandani, H. Benson, and M. Kam, "Robust communication connectivity for multi-robot path coordination using mixed integer nonlinear programming: Formulation and feasibility analysis," in *Proceedings of the International Conference on Robotics and Automation (ICRA 2013)*, Karlsruhe, May 2013.

[11] P. Abichandani, G. Ford, H. Benson, and M. Kam, "Mathematical programming for multi-vehicle motion planning problems," in *Proceedings of the International Conference on Robotics and Automation (ICRA 2012)*, St. Paul, MN, May 2012.

[12] P. Abichandani, K. Mallory, and M. A. Hsieh, "Experimental multi-vehicle path coordination under communication connectivity constraints," in *Proceedings of International Symposium on Experimental Robotics (ISER 2012)*, Quebec City, Canada, June 2012.

[13] P. Abichandani, K. Mallory, and M. A. Hsieh, "Experimental multi-vehicle path coordination under communication connectivity constraints," in *Proceedings of the International Symposium on Experimental Robotics (ISER 2012)*, Quebec City, Canada, June 2012.

[14] 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*, pp. 113–120, 1996.

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

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

[17] 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.

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

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

[20] A. Ben-Tal and A. Nemirovski, "On polyhedral approximations of the second-order cone," *Mathematics of Operations Research*, vol. 26, no. 2, pp. 193–205, 2001. Available from: http://mor.journal.informs.org/content/26/2/193.abstract.

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

[22] H. Y. Benson, "Using interior-point methods within an outer approximation framework for mixed integer nonlinear programming," in *Mixed Integer Nonlinear Programming*, vol. 154, ser. The IMA Volumes in Mathematics and its Applications, (J. Lee and S. Leyffer, eds.), pp. 225–243, New York: Springer, 2012.

[23] 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, Janaury 2010.

[24] 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, pp. 2016–2022, October 2003.

[25] 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*, p. 6, Minneapolis, MN, 2006.

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

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

[28] 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.

[29] 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, pp. 2383–2388, 2002.

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

[31] M. Bussieck and S. Vigerske, "MINLP solver software," *Wiley Encyclopedia of Operations Research and Management Science*, accepted. Available from: http://www.math.hu-berlin.de/~stefan/minlpsoft.pdf.

[32] 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.), pp. 35–59, Springer, 2006.

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

[34] 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.

[35] A. Charles, "New self-driving car system tested on UK roads," Available from: http://www.guardian.co.uk/technology/2013/feb/14/self-driving-car-system-uk.

[36] 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.

[37] 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.

[38] 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*, pp. 4222–4227, Taipei, Taiwan, September 2003.

[39] M. Conforti, G. Cornujols, and G. Zambelli, "Extended formulations in combinatorial optimization," *Annals of Operations Research*, vol. 204, no. 1, pp. 97–143, 2013. Available from: http://dx.doi.org/10.1007/s10479-012-1269-0.

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

[41] 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*, pp. 206–211, November 1991.

[42] 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, September 2009.

[43] 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, pp. 2864–2869, May 1998.

[44] J. Desrosiers, F. Soumis, and M. Desrochers, "Routing with time windows by column generation," *Networks*, vol. 14, no. 4, pp. 545–565, 1984. Available from: http://dx.doi.org/10.1002/net.3230140406.

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

[46] 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.

[47] 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.

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

[49] 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.

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

[51] 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, June 1995.

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

[53] 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, December 2004.

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

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

[56] 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.

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

[58] 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, October 2000.

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

[60] 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.

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

[62] 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*, pp. 2612–2619, 2002.

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

[64] 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.

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

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

[67] 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, December 2002.

[68] G. Inalhan, D. Stipanovic, and C. Tomlin, "Decentralized optimization, with application to multiple aircraft coordination," in *Proceedings of IEEE Conference on Decision and Control*, vol. 1, pp. 1147–1155, 2002.

[69] L. Ingber, *Adaptive Simulated Annealing (ASA)*. Caltech Alumni Association, 1993.

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

[71] 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.

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

[73] 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.

[74] 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, pp. 4921– 4926, Boston, MA, 2004.

[75] 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.

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

[77] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state-dependent graph," *IEEE Transactions on Automatic Control*, vol. 51, no. 1, pp. 116–120, 2006.

[78] G. Knowles, G. W. Haggstrom, T. J. Blaschke, R. Corporation, and U. States, *An Introduction to Applied Optimal Control/Greg Knowles*. New York: Academic Press, 1981.

[79] 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, pp. 1–6, March 1987.

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

[81] J. P. Laumond, "Robot motion planning and control," *Lecture Notes in Control and Information Sciences*, vol. 229.

[82] 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.

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

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

[85] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006. Available from : http://www.planning.cs.uiuc.edu/.

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

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

[88] 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.

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

[90] S. Leyffer, "The return of the active set method," *Oberwolfach Reports*, vol. 2, no. 1, 2005.

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

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

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

[94] 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.

[95] D. Mellinger, A. Kushleyev, and V. Kumar, "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams," in *Proceedings of the International Conference on Robotic and Automation*, St. Paul, MN, May 2012.

[96] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012. Available from: http://ijr.sagepub.com/content/31/5/664.abstract.

[97] 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.

[98] R. E. Moore, "Global optimization to prescribed accuracy," *Computers and Mathematics with Applications*, vol. 21, no. 6/7, pp. 25–39, 1991.

[99] S. Nash and A. Sofer, *Linear and Nonlinear Programming*. New York: McGraw-Hill, 1996.

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

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

[102] G. Nemhauser and L. Wolsey, *Integer and Combinatorial Optimization*. New York: Wiley, 1988.

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

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

[105] 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.

[106] J. Oram, "Governor brown signs California driverless car law at Google HQ," Available from: http://www.brightsideofnews.com/news/2012/9/27/governor-brown-signs-california-driverless-car-law-at-google-hq-.aspx.

[107] 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, March 2002.

[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.

[109] 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.

[110] 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 *Proceedings of MultiRobot Systems Workshop*, pp. 267–278, Kluwer Academic Press, 2003.

[111] A. Piazzi, C. G. L. 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, October 2007.

[112] 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, pp. 915–920, April 1996.

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

[114] 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.

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

[116] C. Reinl and O. von Stryk, "Optimal control of multi-vehicle-systems under comm. constraints using mixed integer linear programming," in *Proceedings of International Conference on Robot Communication and Coordination*, October 2007.

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

[118] 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.

[119] 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, February 1994.

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

[121] T. Schouwenaars, B. DeMoor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *Proceedings of European Control Conference*, pp. 2603–2608, 2001.

[122] 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.

[123] 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.

[124] J. T. Schwartz and M. Sharir, "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.

[125] 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, December 2003.

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

[127] 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.

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

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

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

[131] 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*, pp. 392–397, September 1989.

[132] S. Thrun, "Google's driverless car," Available from: http://on.ted.com/driverless.

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

[134] C. Urmson, "The self-driving car logs more miles on new wheels," Available from: http://googleblog.blogspot.hu/2012/08/the-self-driving-car-logs-more-miles-on.html.

[135] J. van den Berg, S. J. Guy, M. C. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *International Symposium on Robotics Research*, 2009.

[136] J. van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *IEEE International Conference on Robotics and Automation ICRA 2008*, pp. 1928–1935, May 2008.

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

[138] 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, pp. 584–589, September 2003.

[139] N. Vlassis, "Distributed decision making for robot teams," in *Advances in Intelligent and Distributed Computing, Springer Studies in Computational Intelligence*, vol. 78, (C. Badica and M. Paprzycki, eds.), pp. 35–40, 2008.

[140] 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.

[141] 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.

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

[143] 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.

[144] N. Yilmaz, C. Evangelinos, P. F. J. 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, October 2008.