Mathematical Programming for Multi-Vehicle Motion Planning Problems

Pramod Abichandani, Gabriel Ford, Hande Y. Benson, and Moshe Kam

***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, communication connectivity, target tracking, and collision avoidance. The general MVMP problem can thus be formulated as a mathematical program (MP). In this paper we present a mathematical programming (MP) framework that captures the salient features of the general MVMP problem. To demonstrate the use of this framework for the formulation and solution of MVMP problems, we examine in detail four representative works and summarize several other related works. As MP solution algorithms and associated numerical solvers continue to develop, we anticipate that MP solution techniques will be applied to an increasing number of MVMP problems and that the framework and formulations presented in this paper may serve as a guide for future MVMP research.**

I. INTRODUCTION

Multi-Vehicle Motion Planning (MVMP) draws on ideas from Mechanics, Computational Geometry, Algorithmic Decision Theory, Control Theory, and Mathematics. In its most general form, motion planning of a mobile robot is defined as follows: *Given an initial position and orientation and a goal position and orientation of a robot in its workspace, generate a trajectory specifying a continuous sequence of positions, orientations, and speeds while avoiding contact with any obstacles. The trajectory starts at the initial position and orientation and terminates at the goal position and orientation. Report a failure if no such trajectory exists* [1].

This problem, also called the kinodynamic motion planning problem [2], takes into account the kinematics, dynamics, and collision avoidance requirements of the robot and becomes often much more difficult to formulate and solve once extended to coordinated motion of multiple mobile robots amidst multiple stationary and moving obstacles [3], [4], [5]. Coordinated motion planning of multiple robotic vehicles can be used to accomplish several complex tasks with relatively high efficiency. Example applications include automated guidance of ground, underwater, and air vehicles; air traffic control; and manufacturing cell operations.

Mathematical programming frameworks offer the flexibility to accommodate multiple complex constraints simultaneously. Once the constraints have been formulated, an appropriate performance measure can be constructed. In its most general form a mathematical program (1) 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 $\Omega : \mathbb{R}^{n_1} \times \mathbb{Z}^{n_2} \to \mathbb{R}^m$.

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

The difficulty of solving (1) depends on the convexity of the feasible region and objective function, the integrality of the variables, and the size

Gabriel Ford is a doctoral candidate at the Data Fusion Laboratory, Electrical and Computer Engineering Department, Drexel University, Philadelphia, PA 19104, USA gf22@drexel.edu

Pramod Abichandani, Ph.D., is an Assistant Teaching Professor at the College of Engineering, Drexel University, Philadelphia, PA 19104, USA pramod@drexel.edu

Hande Y. Benson, Ph.D., is an Associate Professor at the Decision Sciences Department, Drexel University, Philadelphia, PA 19104, USA hvb22@drexel.edu

Moshe Kam, Ph.D., is the Department Head and Robert Quinn Professor of Electrical and Computer Engineering, Drexel University, Philadelphia, PA 19104, USA kam@minerva.ece.drexel.edu

TABLE I

REPRESENTATIVE WORKS ON MVMP USING MATHEMATICAL PROGRAMMING, INDICATION OF INCLUSION OF KINEMATICS $\mathscr{K}$, DYNAMICS $\mathscr{D}$, COLLISION AVOIDANCE $\mathscr{C}$, COMMUNICATION Θ, AND OTHER CONSTRAINTS $\mathscr{O}$.

Name	MVMP variant	$\mathscr{K}$	$\mathscr{D}$	$\mathscr{C}$	Θ	$\mathscr{O}$	Formulation and Numerical Simulation Solver
Schouwenaars et al. [6] and [7]	Discrete time Decentralized Trajectory planning of multiple UAVs with safety guarantee	✓	✓	✓	-	Hard safety at all times	MILP, `CPLEX`
Peng and Akella [8] and [9]	Discrete space, Continuous time Centralized Kinodynamic Fixed Path Coordination of multiple robots	✓	✓	✓	-	-	MINLP, MILP approximations, `CPLEX`
Derenick et al. [10]	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. [11], [12], [13]	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. [14]	Discrete time Trajectory planning of multiple UAVs via optimal bargaining process	✓	✓	✓	✓	-	NLP, `fmincon`, MATLAB
Richards and How [15]	Discrete time Centralized Trajectory planning of multiple UAVs	✓	✓	✓	-	-	MILP, `CPLEX`, AMPL, MATLAB
Keviczky et al. [16] and [17]	Discrete time Decentralized Receding Horizon Control and Coordination of Autonomous Vehicle Formations	✓	✓	✓	✓	-	MILP, `CPLEX`, AMPL, MATLAB
Pallottino et al. [18]	Discrete time Conflict Resolution for Multiple Aircrafts in a Shared Airspace	-	✓	✓	-	-	MIP, `CPLEX`
Borrelli et al. [19]	Discrete time Centralized Trajectory Generation of Multiple UAVs	✓	✓	✓	-	-	MILP, NLP `CPLEX`, `IPOPT`
Singh and Fuller [20]	Discrete time Centralized Trajectory Generation of a single UAV - extendable to multiple UAVs	✓	✓	-	-	-	QP, `quadprog`, MATLAB
Aoude et al. [21]	Discrete time Multiple Spacecraft Reconfiguration Maneuvers	✓	✓	✓	-	Pointing restrictions	NLP, `SNOPT`

of the problem. When $n_2 = 0$, Φ is a convex function, and $\{\zeta : \Omega(\zeta,\sigma) \leq 0\}$ 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 [22]. Many practical solution algorithms, numerical solvers, and modeling environments have been developed to handle different types of mathematical programs efficiently [22] - [23]. Commercial and/or open-source solvers such as `CPLEX` [24], `SeDuMi` [25], `MOSEK` [26], `LOQO` [27], `IPOPT` [28], `MINLP` [29], `SNOPT` [30], `MATLAB`'s `fmincon` and `quadprog`, and others have been written to solve several MP classes. This is one of the key benefits of using MP as the solvers implementing these algorithms can be directly applied to MVMP problems.

II. REPRESENTATIVE WORKS

The body of work that uses MP techniques to address variants of the general MVMP problem continues to grow [6] - [21], [31]. Table I provides a short introduction to sixteen 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 sixteen studies, we have selected four for a deeper look. These are studies by *Schouwenaars et al.* [6], *Peng and Akella* [8], *Derenick et al.* [10], and by *Abichandani et al.* [11]. Collectively, these four works cover almost all of the features of MVMP problems, viz. kinematics, dynamics, collision avoidance, communication, and visibility/tracking. The problems studied in [6], [8], [10], [11], (as most MVMP problems) can be expressed in the following mathematical program formulation:

$$\begin{aligned}
\textbf{minimize} \quad & \textbf{Objective Function } \Phi(\zeta,\sigma) \\
\textbf{subject to} \quad & \\
& \textbf{Kinematics } \mathscr{K}(\zeta,\sigma) \leq 0 \\
& \textbf{Dynamics } \mathscr{D}(\zeta,\sigma) \leq 0 \\
& \textbf{Collision-avoidance } \mathscr{C}(\zeta,\sigma) \leq 0 \\
& \textbf{Communication } \Theta(\zeta,\sigma) \leq 0 \\
& \textbf{Other Constraints } \mathscr{O}(\zeta,\sigma) \leq 0
\end{aligned} \tag{2}$$

Schouwenaars et al. in [6] present a cooperative decentralized algorithm for trajectory generation of multiple UAVs with hard safety guarantees. Of main interest in this work 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). The work presented in this paper led to one of the first practical implementations of MP based motion planning of UAVs [7].

In [8], *Peng and Akella* deal with the problem of collision-free coordination of multiple robots with constraints on kinematics and dynamics along specified paths such that the traversal time of the set of robots is minimized. The solution of this problem is a time schedule for each robot along its path. Each robot's path is divided into segments. Each segment is then checked for a possibility of collisions and is accordingly labelled as collision segment or collision-free segment. The authors use a well established result obtained from optimal control theory (minimum and maximum time control of a double integrator with inequality constraints on control (acceleration) and state (velocity and hence position) [32]) to obtain closed form formulae for minimum and maximum times that the double integrator should take to traverse a path segment of given length. Using these closed form expressions as constraints on robot path segment traversal times, the authors construct a Mixed Integer Nonlinear Program (MINLP) with additional constraints on robot vehicle kinematics, dynamics, and collision avoidance.

In [10], *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) [33] involving Laplacian matrices of graphs that represent target visibility and inter-robot communication connectivity. This work demonstrates the use of spectral graph theory and semidefinte programming techniques to solve the target tracking problem.

In [11], 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 up to 50 vehicles.

III. General Mathematical Programming Structure of MVMP problems

As we stated, in its most general form, a MVMP problem can be expressed by the Mathematical Program (2) and [6], [8], [10], and [11] are excellent examples of the use of (2). In the following, we highlight details about the formulations and results discussed in these four studies.

A. Decision Variables ζ, σ

Depending on the task to be performed and modeling approach adopted, the decision variables for (2) can be reference speed of each vehicle [6], segment traversal times and speeds at the start of each segment of a discretized path space [8], positions of the robot vehicles in continuous Euclidean space [10], or speed along the fixed paths represented by cubic spline path primitives [11]. For centralized formulations in [8], [10], [11], the values of all decision variables are determined simultaneously. For decentralized formulations presented in [6], [13] each vehicle determines the values of its own speed in a sequential manner.

B. Objective Function $\Phi(\zeta, \sigma)$

In the decentralized formulation of *Schouwenaars et al.* [6], $\zeta = \mathbf{u_k}$, where $\mathbf{u_k}$ is the reference speed of each vehicle at current time step $\mathbf{k}$, and the objective function for each vehicle $\Phi_a(\zeta)$ is a piecewise linear cost function that comprises of the weighted 1-norm of the difference of the current state $\mathbf{x_k}$ with the desired state $\mathbf{x_f}$ minus the scalar product of the current velocity vector $\mathbf{s_k}$ with the vector indicating the direction from initial position $\mathbf{p_0}$ to the final position $\mathbf{p_f}$. The values of $\mathbf{x_k}$ and $\mathbf{s_k}$ depend on the decision variable $\zeta = \mathbf{u_k}$. q and r are weights that can be tuned appropriately. $\Phi_a(\zeta)$ provides a minimum time formulation.

$$\Phi_a(\zeta) = \sum_{k=1}^{T} [(q'|\mathbf{x_k} - \mathbf{x_f}|) - r(\mathbf{p_f} - \mathbf{p_0})' \mathbf{s_k}], \qquad (3)$$

In [8], the scenario completion time, which is maximum time taken by the last arriving robot, is minimized. To maximize the tracking visibility, *Derenick et al.* in [10] minimize the negative of the second smallest eigenvalue of the visibility graph.

The centralized formulation in [11] is a minimum time formulation with $\zeta = \mathbf{s_k}$, where $\mathbf{s_k}$ is a vector of (continuous valued) speeds of all robots along their paths at discrete time step $\mathbf{k}$. The formulation minimizes $\Phi_b(\zeta)$, where

$$\Phi_b(\zeta) = \mathbf{T}\mathrm{max} + \gamma \sum_{\mathbf{i},\mathbf{k}} \mathbf{d^i_{goal}}(\mathbf{k}) \qquad (4)$$

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

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

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

C. Kinematic Constraints $\mathscr{K}(\zeta, \sigma) \leq 0$

To represent vehicle kinematics, *Schouwenaars et al.* in [6] use a discretized velocity control model, *Peng and Akella* use a double integrator model in [8], *Derenick et al.* use a single order fully actuated model [10]. In [11], piecewise cubic spline functions with continuous first and second derivatives are used to represent the fixed paths. The spline curve $ps^{\mathbf{i}}(u^{\mathbf{i}}(\mathbf{k}))$ defines the position $(x^{\mathbf{i}}(\mathbf{k}), y^{\mathbf{i}}(\mathbf{k}))$ for each robot $\mathbf{i}$ at time step $\mathbf{k}$. $u^{\mathbf{i}}(\mathbf{k})$ is the arc-length traversed by the robot $\mathbf{i}$ along its spline path at time $\mathbf{k}$. $u^{\mathbf{i}}(\mathbf{k})$ depends on the speed $s^{\mathbf{i}}(\mathbf{k})$ and time-step Δt (5).

$$\begin{aligned} (x^{\mathbf{i}}(\mathbf{k}), y^{\mathbf{i}}(\mathbf{k})) &= ps^{\mathbf{i}}(u^{\mathbf{i}}(\mathbf{k})) \\ u^{\mathbf{i}}(0) &= 0 \\ u^{\mathbf{i}}(\mathbf{k}) &= u^{\mathbf{i}}(\mathbf{k}-1) + s^{\mathbf{i}}(\mathbf{k})\Delta t \end{aligned} \qquad (5)$$

D. Dynamic Constraints $\mathscr{D}(\zeta,\sigma) \le 0$

Schouwenaars et al. in [6], linearize the non-linear dynamic bounds by using N-sided polygonal approximations of a circle that represents dynamic bounds on speed. Non-linear bounds on speed and acceleration are used in [8] and [10] (second order conic inequalities). In [11], 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.

E. Collision Avoidance Constraints $\mathscr{C}(\zeta,\sigma) \le 0$

Collision avoidance constraints require the robots to avoid colliding with obstacles as well as other robots in the workspace at all times. In [6], *Schouwenaars et al.* represent collision avoidance using loiter circles. In [10], *Derenick et al.* use conditions on Euclidean Distance Matrix (EDM) [34] to model collision avoidance constraints while in [11], 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 [8], 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 could potentially collide if simultaneously traversing segments z_i and z_j respectively, then one of the following must hold to avoid collisions:

$$t^j_{z_j} \ge t^i_{z_i+1} \tag{6}$$

$$t^i_{z_i} \ge t^j_{z_j+1} \tag{7}$$

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 can be converted into (8).

$$\begin{aligned} t^j_{z_j} - t^i_{z_i+1} + M(1-\delta^{ij}_{z_i z_j}) \ge 0 \\ t^i_{z_i} - t^j_{z_j+1} + M\delta^{ij}_{z_i z_j} \ge 0 \\ \delta^{ij}_{z_i z_j} \in \{0,1\} \end{aligned} \tag{8}$$

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

where M is an arbitarily large number [35].

F. Communication Constraints $\Theta(\zeta,\sigma) \le 0$

Communication requirements are mostly specified in terms of connectivity between the vehicles. While [6] assumes that there is a centralized communication hub that takes care of all communication needs, [10] and [11] explicitly deal with these communication requirements. In [10], *Derenick et al.* use constraints on the Laplacian matrix of a communication graph to express communication connectivity requirements. Given a set of nodes $\mathscr{V}_n$ and a set of edges $\mathscr{E}_m$, let $G(\mathscr{V}_n,\mathscr{E}_m)$ be a weighted graph with n vertices and m edges. The Laplacian $L(G)$ of G is a n-dimensional square matrix whose entries are denoted by

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

where, $w_{\psi\alpha}$ is the weight associated with the edge between vertices ψ and α. There are certain important properties of $L(G)$ that make it particularly useful for several applications. Specifically, the communication connectivity constraint states that the communication connectivity graph should be connected at all times and hence the transformation $P_N^T L P_N$ should be positive definite, where P is a n x $(n-1)$ matrix [36].

In [11], we use the Friis's free-space communication model to capture the propagation and power loss in the radio communication channels [37]

$$P_r = P_{tr} G_t G_r \left(\frac{\lambda}{4\pi d}\right)^{\alpha} \tag{9}$$

where α is the path loss exponent. P_r is the received power at the receiver and P_{tr} is the power at which the signal was transmitted. The noise is assumed to be thermal ($kTBF$). λ is the wavelength and is equal to c/f, where c = 3 x 10^8 m/s and f = 2.4 x 10^9 Hz. The values of G_t and G_r (antenna gains) is assumed to be 1. The values of α range from 1.6 (indoor with line of sight) to 6 (outdoor obstructed) depending on the environment.

G. Other Constraints $\mathscr{O}(\zeta,\sigma) \leq 0$

In [10], *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 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 [36].

IV. Observations and Design Considerations

A. Generality

1) Concepts/results from other disciplines: The framework is general enough to incorporate concepts and results from other disciplines. In [6], the mathematical program is embedded in a decentralized decision-making algorithm. In [8], the authors make use of results from optimal control theory to get bounds on the decision variables, while in [10], the authors use results from spectral graph theory to express visibility and communication constraints. In [11], 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 [21], *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 [38]. The output of the first stage acts as an initial guess to second stage NLP formulations.

2) Discretization of Space and/or Time: This framework is general enough to allow discretization of space and/or time. Studies [6], [10], and [11] are examples of continuous space and discrete time formulations. In [8], 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 [32].

3) Path Primitives: This framework is general enough to allow any path primitives such as straight lines and circular arcs [8], two dimensional cubic splines[11] among others to be used.

4) Centralized/Decentralized Decision Making: The framework allows for centralized decision making schemes as witnessed in case of [8], [10], and [11] or decentralized decision making schemes, wherein multiple decision makers independently solve their optimization problems [6], [13].

5) Receding Horizon/Model Predictive Control: If computational resources are limited and/or complete information is unavailable, instead of solving a problem for all time steps for all vehicles, one can use a receding horizon/model predictive approach to solve similar problems over multiple time steps [6], [20].

B. Advantages accorded by MP

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

Due to the fact that certain nonlinear constraints add non-convexity to the problem, many models are constructed by linearizing/approximating the nonlinearities and encoding non-convexities using integer variables. This approach enables the use of powerful integer programming solvers. This can be seen in [6], where the authors develop polygonal approximations of a circle to express dynamic constraints. On the other hand, the authors in [39] 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, nonconvex constraints [23]. Under certain conditions, lack of convexity due to discrete variables can be handled using branch and bound and cutting plan algorithms for MILP, and branch and bound and outer approximation algorithms for MINLP [23].

2) Modeling Environments, Solution Algorithms and Solvers: Modeling environments such as `AMPL` [40], `GAMS` [41], and `YALMIP` [42] can be used for rapid development of models for testing/simulation purposes. *Schouwenaars et al.* in [6] implement their entire framework using `AMPL`. In particular, several large-scale NLP solution techniques can handle non-convexities but do not provide guarantees of finding globally optimal solution. These include solvers like `LOQO` that uses interior-point methods [27], `KNITRO` that uses trust-region algorithms [43], and `SNOPT` that uses a quasi-Newton algorithm [30] among others. These solvers also incorporate mechanisms to detect infeasibilities and unboundedness in the problem. Many solvers heavily preprocess the problem before proceeding to solving them thereby speeding up the solution process tremendously. `CPLEX` [24] is one such commercial solver that has been successfully used to solve problems practical MVMP problems in real-time [7], [31]. Furthermore, in recent work, highly efficient code has been written to implement these algorithm in real time [44], [45].

V. Conclusion

In this paper we reviewed several publications that use MP to solve variants of the general MVMP problem. We presented a general mathematical programming based framework that can accommodate all objectives and constraints of a MVMP problem, and focused on four representative efforts that serve as examples of using this framework. 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, receding horizon/model predictive control in a vast array of real-world settings. In addition, the framework lends itself to numerical solutions using readily available commercial and/or open-source solvers and in many cases via efficient polynomial time algorithms.

While these advantages certainly make MP an attractive option to solve MVMP problems, we would like to highlight the fact that a prohibitively large number of decision variables and constraints in an MP problem can result in increased solution times, and solutions that are not necessarily globally optimal. However, 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.

References

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

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

[3] J. Hopcroft, J. Schwartz, and M. Sharir, “On the complexity of motion planning for multiple independent objects: PSPACE-hardness of the “warehouseman’s problem”,” *Int. J. of Rob. Research*, vol. 3, no. 4, pp. 76–88, 1984.

[4] J. H. Reif, “Complexity of the mover’s problem and generalizations,” in *Proc. IEEE Symposium on Foundations of Computer Science, 1979*.

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

[6] 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.

[7] T. Schouwenaars, “Safe trajectory planning of autonomous vehicles,” *PhD thesis, MIT, 2005*.

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

[9] J.Peng and S. Akella, “Coordinating multiple double integrator robots on a roadmap: Convexity and global optimality,” in *Proc. IEEE Int. Conf. on Robotics and Automation, 2005*.

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

[11] P. Abichandani, H. Benson, and M. Kam, “Multi-vehicle path coordination in support of communication,” in *Proc. Int. Conf. on Robotics and Automation, 2009*, Kobe, Japan.

[12] ——, “Multi-vehicle path coordination under communication constraints,” in *Proc. American Control Conf., 2008*.

[13] ——, “Decentralized path coordination in support of communication,” in *Proc. Int. Conf. on Robotic Systems*, San Francisco, CA, 2011.

[14] G. Inalhan, D. Stipanovic, and C. Tomlin, “Decentralized optimization, with application to multiple aircraft coordination,” in *Proc. of the 41st IEEE Conf. on Decision and Control*, vol. 1, 2002, pp. 1147 – 1155.

[15] A. Richards and J. How, “Aircraft trajectory planning with collision avoidance using mixed integer linear programming,” in *Proc. of American Control Conf., 2002*.

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

[17] T. Keviczky, F. Borrelli, and G. Balas, “A study on decentralized receding horizon control for decoupled systems,” in *Proc. of American Control Conf., 2004*.

[18] L. Pallottino, E. Feron, and A. Bicchi, “Conflict resolution problems for air traffic management systems solved with mixed integer programming,” *IEEE Tran. on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 3 –11, Mar. 2002.

[19] F. Borrelli, D. Subramanian, A. Raghunathan, and L. Biegler, “MILP and NLP techniques for centralized trajectory planning of multiple unmanned air vehicles,” in *Proc. of American Control Conf., 2006*.

[20] L. Singh and J. Fuller, “Trajectory generation for a UAV in urban terrain, using nonlinear MPC,” in *Proc. of American Control Conf., 2001*.

[21] G. Aoude, J. How, and I. Garcia, “Two-stage path planning approach for solving multiple spacecraft reconfiguration maneuvers,” *AAS J. of Astro. Sci.*, vol. 56, no. 5, pp. 515 –544, 2008.

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

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

[24] Ilog, Inc., “Solver cplex.” [Online]. Available: http://www.ibm.com/software/integration/optimization/cplex-optimizer/ (accessed 19 March 2011)

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

[26] “The MOSEK optimization software.” [Online]. Available: http://www.mosek.com/

[27] R. Vanderbei, “LOQO user’s manual—version 3.10,” *Optimization Methods and Software*, vol. 12, pp. 485–514, 1999.

[28] 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.

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

[30] 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.

[31] A. Richards, Y. Kuwata, and J. How, “Experimental demonstrations of real-time MILP control,” in *Proc. of the AIAA Guidance, Navigation, and Control Conf. 2003*.

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

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

[34] J. Gower, “Properties of euclidean and non-euclidean distance matrices,” *Linear Algebra and its Applications*, pp. 81–97, June.

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

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

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

[38] I. Garcia and J. How, “Trajectory optimization for satellite reconfiguration maneuvers with position and attitude constraints,” in *Proc. of American Control Conf., 2005*, pp. 889 – 894 vol. 2.

[39] 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.

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

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

[42] J. Löfberg, “YALMIP : A toolbox for modeling and optimization in MATLAB,” in *Proceedings of the CACSD Conference*, 2004.

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

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

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