Mixed-Integer Nonlinear Programming Formulation of a UAV Path Optimization Problem

Shankarachary Ragi and Hans D. Mittelmann

***Abstract*— We present a mixed-integer nonlinear programming (MINLP) formulation of a UAV path optimization problem, and attempt to find the global optimum solution. As objective functions in UAV path optimization problems tend to be non-convex, traditional optimization solvers (typically local solvers) are prone to local optima, which lead to severely sub-optimal controls. For the purpose of this study, we choose a target tracking application, where the goal is to optimize the kinematic controls of UAVs while maximizing the target tracking performance. First, we compare the performance of two traditional solvers numerically - MATLAB's *fmincon* and *knitro*. Second, we formulate this UAV path optimization problem as a *mixed-integer nonlinear program* (MINLP). As this MINLP tends to be computationally expensive, we present two pruning methods to make this MINLP tractable. We also present numerical results to demonstrate the performance of these methods.**

Index Terms*— UAV path optimization, target tracking, mixed-integer nonlinear programming, *fmincon*, *knitro

I. INTRODUCTION

Path planning algorithms for unmanned aerial vehicles (UAVs) are gaining importance owing to their applications in target tracking [1], surveillance [2], reconnaissance [3], etc. With this motivation, in the past, we have developed UAV guidance algorithms for target tracking applications using *partially observable Markov decision process (POMDP)* and *decentralized POMDP* frameworks [4], [5], [6]. Irrespective of the applications, the formulations of UAV guidance problems typically boil down to solving a non-convex optimization problem, where finding the optimal solution is challenging. For instance, the UAV guidance problems in [4], [6] too required minimizing a cost function, which is non-convex.

To tackle non-convex objective functions in the context of UAV path optimization, efficient optimization methods are being studied. But in most instances in the literature, mixed-integer linear programs were used to solve path planning problems, e.g., [7] and [8]. In most real-world applications, the objective functions tend to be nonlinear and non-convex, and it seems there are no existing studies in the literature that provide tractable solutions to mixed-integer nonlinear programs (MINLPs) in the context of UAV path optimization. In our previous study, we solved an MINLP that resulted from a UAV guidance problem [4] via MATLAB's *fmincon* (a local solver), which may have lead to suboptimal controls. With the motivation to find the global optimum solution to the above-mentioned MINLP, we adopt a UAV path planning problem for a target tracking application considered in [4], [9]; but the method presented in our study can be extended to other applications where the objective function is non-convex. We do not guarantee that our proposed method achieves the global optima for any generic *non-convex* optimization problem; we only attempt to study the performance of the MINLP formulation in the context of UAV path optimization problems.

We adopt a target tracking application, where the objective is to control the motion of UAVs to track moving targets. Without loss of generality, and for simplicity, we assume that the targets and the UAVs move in 2-D as in [4]. We adopt the kinematic models presented there to describe the motion of a UAV and a target. The UAVs are equipped with sensors that generate noisy observations of the target locations. We are not concerned with the internal processes of a sensor. We simplify the sensor process by assuming that the observations of the target locations are available (bypassing the step where the location is extracted from the measured image frame if the sensor is a camera), which are corrupted by zero-mean Gaussian noise. The characteristics of measurement noise in the observation of the location of a target at a sensor depends on the relative location of the target with respect to the sensor (or UAV on which the sensor is mounted). The location of a UAV can be controlled by the kinematic controls, specifically its *lateral acceleration* and *bank angle*. We adopt the kinematic model for UAV motion dynamics presented in [4]. As the target locations and their velocities are not known exactly (measurements are corrupted by noise), we estimate these parameters using the measurements, measurement model, and the target motion model. The objective is to optimize the UAV controls to minimize the *mean-squared error* between the target state and the target state estimate over a time horizon. In this study, we solve the UAV path planning problem via *mixed-integer nonlinear programming (MINLP)* in an attempt to find the globally optimum solution. Although we restrict the movement of UAVs to 2-D, the methods we present in this study can be easily extended to 3-D, but it may result in increased computational intensity.

Mixed-Integer Programming (MIP) formulation was considered for UAV guidance problems for various applications. A UAV path optimization problem was formulated as a *mixed-integer linear program (MILP)* and solved via CPLEX solver in [10], where the task-allocation and the trajectory optimization (way-point optimization) were addressed. In

This work was supported in part by Air Force Office of Scientific Research under grant FA 9550-15-1-0351.

S. Ragi and H. D. Mittelmann are with the School of Mathematical and Statistical Sciences, Arizona State University, Tempe, AZ 85287-1804, USA {shankarachary.ragi, mittelmann}@asu.edu

[11], a UAV monitoring-routing problem was addressed via MIP formulation and solved via CPLEX, where a time-space network model was used to optimize UAV revisit times. A UAV task assignment problem was formulated as an MILP in [12]. Although MIP formulations have been used for UAV task assignment problems (see references above and [13], [14], [15]), exact methods for optimization of UAV kinematic controls (e.g., bank angle and lateral acceleration) for target tracking have not been studied. In this study, we present the MINLP formulation of the above-mentioned UAV path optimization problem, and more importantly we develop certain strategies to make the MINLP tractable. This method is inspired from [16], where exact methods were explored to solve a combinatorial optimization problem in the context of directional sensor control.

The UAV path optimization problem studied here falls in the category of *model-predictive control* (MPC) problems (see [4] for details). Exact methods to solve MPC problems were considered in [17], [18]. Our study enhances this existing literature, as our work is an extension to [4], where the UAV guidance problem was posed as a *partially observable Markov decision process* (POMDP), which is a generalization of MPC [4].

II. PROBLEM SPECIFICATION

For completeness, we describe the UAV guidance problem in [4] briefly. Let k be the time index, and $x_k = (s_k, \chi_k, \xi_k, P_k)$ represent the state of the system at time k. Here, s_k denotes the state of the UAVs including their locations and velocities and χ_k is the state of the targets including their locations, velocities, and accelerations. The state of the tracking algorithm (Kalman filter) is denoted by (ξ_k, P_k), where ξ_k is the posterior mean vector and P_k is the posterior covariance matrix. The vector u_k includes the lateral acceleration and bank angle controls of the UAVs. The UAV state vector s_k evolves according to $s_{k+1} = \psi(s_k, u_k)$, where ψ being the UAV kinematic motion model given the control vector u_k (see [4] for the definition of ψ). The target state χ_k evolves according to the *constant velocity* model [4] as shown below:

$$\chi_{k+1} = \mathbf{F}\chi_k + v_k,\ v_k \sim \mathcal{N}(0, \mathbf{Q}),$$

where $\mathbf{F}$ denotes the state transition model and $\mathbf{Q}$ the process covariance matrix. The observations of the target locations at a sensor (mounted on a UAV) is given by:

$$z_k = \mathbf{G}\chi_k + w_k,\ w_k \sim \mathcal{N}(0, \mathbf{R}_k(\chi_k, s_k)),$$

with $\mathbf{G}$ being the state transition model and $\mathbf{R}_k(\chi_k, s_k)$ the measurement error covariance matrix that depends on the relative location of the target with respect to the sensor (UAV). The state of the tracking algorithm (ξ_k, P_k) evolves according to Kalman filter equations. The objective is to plan the control actions $u_k = (a_k, \phi_k)$, $k = 1, \ldots, H$ over a time horizon of length H such that the cumulative sum (over horizon H) of the mean-squared errors between the target state estimates and the target states is minimized, where a_k and ϕ_k capture the lateral acceleration and bank angle controls respectively for the UAVs at time k.

III. METHODS

In [4], [9], the UAV guidance problem described in the previous section was posed as a POMDP. Since POMDPs are hard to solve exactly, an approximation method called *nominal belief state optimization* (NBO) was introduced to solve the UAV guidance problem approximately. The POMDP formulation is not presented here as it is beyond the scope of this paper. The UAV guidance problem in [4], after NBO approximation, results in the following optimization problem:

$$\begin{aligned}
\underset{u_k;\ j=1,\ldots,H}{\text{minimize}} \quad & \sum_{k=1}^{H} \text{trace}(A_k^{-1}) \\
\text{subject to} \quad & A_k = P_{k|k-1}^{-1} + G^T R_k^{-1} G;\ \forall k = 1, \ldots, H \\
& P_{k|k-1} = F A_{k-1}^{-1} F^T + Q,\ \forall k = 1, \ldots, H \\
& u_k = (a_k, \phi_k), a_k \in [-a_{\max}, a_{\max}], \phi_k \in [-\phi_{\max}, \phi_{\max}] \\
& \forall k = 1, \ldots, H
\end{aligned} \tag{1}$$

where R_k is the target measurement covariance matrix at time k, which depends on the locations of the UAVs and the targets at time k, which in-turn are a function of the controls u_k, $k = 1, \ldots, H$, and $P_k = A_k^{-1}$, $k = 1, \ldots, H$ are the state covariance matrices. The non-convex dependence of the measurement covariance matrix on the controls u_k makes it hard to achieve the global optimum with standard solvers. See [4] for the definition of the measurement covariance matrix, which shows the non-convex dependence of the measurement covariance matrix on the UAV kinematic controls. In [4], MATLAB's *fmincon* was used to solve this optimization problem, which is prone to local optima. As the above problem is non-convex, our study is focused on checking if the MINLP formulation can potentially achieve the global optimum.

A. Fmincon vs. Knitro

To verify if *fmincon* is indeed susceptible to local optima, we compare the performance of *fmincon* with another commercial solver called *knitro* for 50 different initial solutions. Without loss of generality, we implement a scenario with a single UAV and a single target. We set the horizon length H to 10 steps. In (1), we set $a_{\max} = 5$ and $\phi_{\max} = 5$. We solve this problem instance with both solvers *fmincon* and *knitro*. Figure 1 shows that *knitro* outperforms *fmincon* by a clear margin, which suggests that the *fmincon* solver is not achieving the global optimum. Although *knitro* tends to outperform *fmincon*, but it too does not guarantee a global optimum. Therefore, in the following subsections, we attempt to develop an exact method based on MINLP to potentially find the global optimum for this UAV path optimization problem.

The solver *knitro* has a feature called *multi-start*, which when set to a certain numerical value n_{MS}, the solver finds

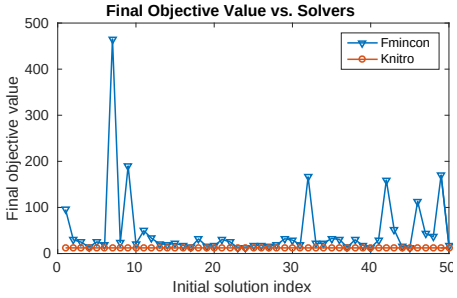

Fig. 1. Final objective value vs. initialization index

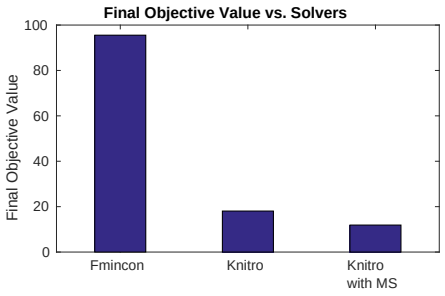

Fig. 2. Final objective value comparison: *fmincon* vs. *knitro*

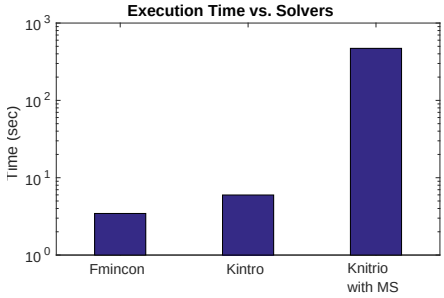

Fig. 3. Execution time: *fmincon* vs. *knitro*

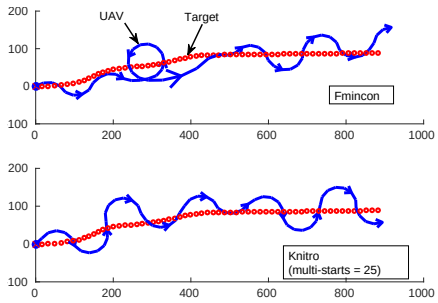

Fig. 4. Comparing UAV trajectories: *fmincon* vs. *knitro*

the best solution from n_{MS} different initializations. For the purpose of this study, we set $n_{MS} = 10$. We compute the final objective values from each of the methods - *fmincon*, *knitro*, and *knitro with multi-start*. Figures 2 and 3 show the performance comparison of *fmincon*, *knitro*, and *knitro with multi-start* in solving the continuous optimization problem described in (1). Clearly, the solver *knitro (with and without multi-start)* outperforms MATLAB's *fmincon*.

Table I shows the final objective values with increasing n_{MS} (described above) with the *knitro* solver. The results from this table show a trend - the final objective value gets saturated for $n_{MS} \geq 5$, which appears to suggest that *knitro* is able to find the global optimum with just 5 multi-starts.

We now compare the performance of *knitro* and *fmincon* in a dynamic setting, described as follows. A UAV is tracking a single target, where at each time step we plan (by solving (1) using *knitro* or *fmincon*) the control actions for the UAV over a planning horizon (for this case, we set $H = 6$), implement the UAV control action for the current time-step from the planned control actions, and discard the control actions corresponding to the future time steps. We repeat this every time-step. In other words, we implement a *receding horizon control* approach. Figure 4 shows the trajectories of a UAV tracking a single target using solvers *fmincon* and *knitro*. For a quantitative comparison of the performance, we run a Monte Carlo simulation, where we simulate the above dynamic scenario for 100 runs. In each run, we evaluate the average target-location error (the average is over the run-time of the simulation) for each solver independently. Figure 5 shows the cumulative distribution function (CDF) plots for both *fmincon* and *knitro*. From the CDF plot, it is evident that *knitro* has a statistical edge over *fmincon* in performance with respect to the average target-location error metric.

TABLE I

PERFORMANCE OF *knitro* VS. NO. OF MULTI-STARTS

Solver	Final Objective Value
Fmincon	95
Knitro with $n_{MS} = 1$	18.07
Knitro with $n_{MS} = 3$	15.08
Knitro with $n_{MS} = 5$	12.2
Knitro with $n_{MS} = 10$	12.2
Knitro with $n_{MS} = 20$	12.2

B. MINLP with Pre-Computations

From the results in the previous subsection, it was evident that *fmincon* is susceptible to local optima. With the intent to develop an exact method, which finds the global optimum, we now pose the UAV path optimization problem as a *mixed-integer nonlinear program* (MINLP). First, we discretize the action space, i.e., discretize the lateral acceleration interval and the bank angle interval. Let $A = \{a_1, \ldots, a_M\}$ be a discrete set of lateral accelerations, and $\Phi = \{\phi_1, \ldots, \phi_N\}$ be a discrete set of bank angles that can be enforced on the UAV at any time-step. Therefore, $U = \{(a, \phi) | a \in A, \phi \in \Phi\}$ is the set of all possible controls available at each time-step; note that there are MN elements in U.

Our cost function depends on the target measurement covariance matrices R_k, $k = 1, \ldots, H$, which depend on the control actions over the planning horizon H. The cost function is non-convex because of the non-convex dependence of the target measurement covariance matrices on the control actions (see [4] for details). Therefore, in an attempt to remove this non-convexity, we change the problem formulation and pose it as a *mixed-integer nonlinear program*.

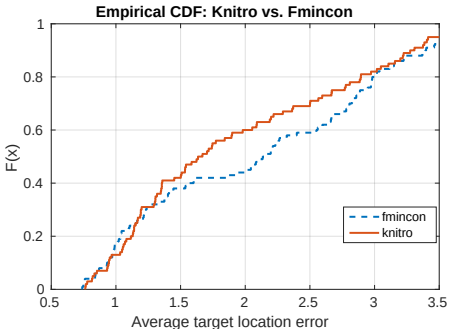

Fig. 5. Quantitative performance comparison of *fmincon* and *knitro* in a dynamic setting

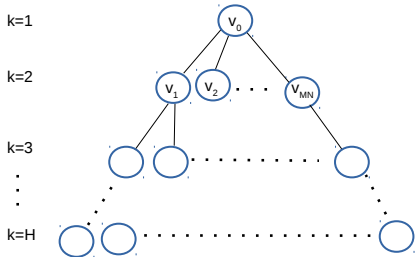

Fig. 6. Branching model to represent all possible control actions (and associated measurement covariance matrices) over the time-horizon H

First, we pre-compute all possible target measurement covariance matrices from all possible control actions over the planning horizon H. We adopt a branching method to find the measurement covariance matrices corresponding to all possible control sequences. We assume a tree model with nodes and branches, where a node at level k represents a particular sequence of actions planned until time k. At each node in the tree, we branch out to MN possible child nodes, as there are MN possible actions available at each step. To each node, we also associate a covariance matrix, which denotes the target measurement covariance matrix evaluated with the sequence of controls associated with this node. Let us associate a binary variable to each node given by v_i, $i = 1, \ldots, \sum_{k=1}^{H} (MN)^k$, where $v_i = 1$ implies we include the action v_i in our action plan, and $v_i = 0$ otherwise. This tree model with the associated binary variables is depicted in Figure 6.

Clearly, the sum of the child binary variables under each parent node in the tree should sum up to 1 if the parent binary variable is 1, and should sum to 0 otherwise, i.e.,

$$\sum_{j=(MN)i+1}^{(MN)(i+1)} v_j = v_i, \forall\, i = 0, \ldots, \sum_{k=1}^{H-1} (MN)^k,$$
$$\text{and } v_0 = 1.$$

We pre-compute all possible target measurement covariance matrices $R_1, \ldots, R_{\sum_{k=1}^{H}(MN)^k}$ from every possible action sequence for each time-step in the planning horizon, e.g., $R_1, \ldots, R_{MN}$ are MN possible target measurement covariance matrices from MN possible actions for the first time-step. We now pose the UAV guidance problem as a *mixed-integer nonlinear program* as follows:

$$\underset{v_j;\ j=1,\ldots,\sum_{k=1}^{H}(MN)^k}{\text{minimize}} \quad \sum_{i=1}^{H} \text{trace}(A_i^{-1})$$
$$\text{subject to} \quad A_i = P_{i|i-1}^{-1} + G^T \left(\sum_{j=q_i}^{q_{i+1}-1} R_j^{-1} v_j \right) G;\ \forall i = 1, \ldots, H$$
$$P_{i|i-1} = F A_{i-1}^{-1} F^T + Q,\ \forall i = 1, \ldots, H$$
$$q_i = \sum_{k=0}^{i-1} (MN)^k,\ \forall i = 1, \ldots, H$$
$$\sum_{j=(MN)i+1}^{(MN)(i+1)} v_j = v_i,\ \text{forall } i = 0, 1, \ldots, \sum_{k=1}^{H-1} (MN)^k,\ \text{and } v_0 = 1$$
$$v_j = \{0, 1\},\quad \forall j.$$

Note that with the above formulation, we removed any non-convexity of the objective function (due to the non-convex dependence of the target measurement covariance matrix on the UAV controls) via pre-computations. Although, the above problem can be solved exactly, the pre-computations required grow exponentially with the planning horizon length H, and also the number of binary variables $v_j\ \forall j$ grows exponentially with the horizon length. The performance of this method depends on the resolution of the discretization of the control space. If the resolution is too low, this method may not find the global optimum, and may even perform worse than a local solver (e.g., MATLAB's *fmincon*) for the continuous optimization problem. However, if the resolution is too high, the computational time required to solve the MINLP increases. Therefore, one may need to trade off between the ability to find the global optimum and the computation time, which is a usual trade off in any non-convex optimization problem. In the next subsection, we present a method to keep the pre-computations and the computational time to solve the MINLP from growing exponentially with the length of the horizon H.

C. MINLP with Pre-Computations and Pruning

We limit the number of maximum possible pre-computed measurement covariance matrices at each time step, i.e., we limit the number of nodes in the tree (described in the previous subsection) to a maximum of K at each level. Therefore, we pre-compute KH possible measurement covariance matrices. We present two methods for pruning called PrunMethod-A and PrunMethod-B as described below.

PrunMethod-A is described in the following steps: 1) Given K selected measurement covariance matrices from the previous planning step, we generate $K(MN)$ measurement covariance matrices represented by $R_{K1}, R_{K2}, \ldots, R_{K(MN)}$; 2) Rearrange these $K(MN)$ matrices in an order with increasing value of their traces, and let the rearranged sequence be $R_{Kr_1}, R_{Kr_2}, \ldots, R_{Kr_{MN}}$; 3) Retain the first K matrices $R_{Kr_1}, \ldots, R_{Kr_K}$ and discard the remaining matrices; 4) Repeat this in each time-step up to H. A pseudo-code for this method is shown in Algorithm 1.

PrunMethod-B is described in the following steps: 1) Given the tree of measurement covariance matrices, we

associate a cost for each node called *branch-cost*, which is the sum of traces of measurement covariance matrices of the branch starting from root node to the current node; 2) Given K selected measurement covariance matrices from the previous planning step, we select $K(MN)$ measurement covariance matrices which are the child nodes of the K parent nodes represented by $R_{K1}, R_{K2}, \ldots, R_{K(MN)}$; 3) Rearrange these $K(MN)$ matrices in an order with increasing value of the weighted average of the trace of the measurement covariance matrix for the current node and *branch-cost* of its parent node, and let the rearranged sequence be $R_{Kr_1}, R_{Kr_2}, \ldots, R_{Kr_{MN}}$; 4) Retain the first K matrices $R_{Kr_1}, \ldots, R_{Kr_K}$ and discard the remaining matrices; 5) Repeat this in each time-step up to H. A pseudo-code for this method is shown in Algorithm 2.

Algorithm 1 Prune Method A

$k \leftarrow 2$ (time index)
$K \leftarrow$ max. allowed nodes in each level
$H \leftarrow$ time-horizon
$G \leftarrow$ sequence of measurement covariance matrices from all possible controls at initial time-step
$G \leftarrow$ Rearrange(G) ▷ Rearranges the sequence in the order of increasing trace values
$G \leftarrow$ Prune(G, K) ▷ prunes the sequence by picking the first K elements
loop:
if $k < H$ **then**
    $G \leftarrow$ GenMeas(G) ▷ generates the sequence of covariance matrices at level k given matrices from level $k-1$
    $G \leftarrow$ Rearrange(G)
    $G \leftarrow$ Prune(G, K)
$i \leftarrow i + 1$.
goto *loop*.

Algorithm 2 Prune Method B

$k \leftarrow 2$ (time index)
$K \leftarrow$ max. allowed nodes in each level
$H \leftarrow$ time-horizon
$G \leftarrow$ sequence of measurement covariance matrices from all possible controls at initial time-step
$C_G \leftarrow$ WgtAvg(G) ▷ associates a real value to each element in G: weighted average of trace of the element matrix and the branch cost of its parent node
$G \leftarrow$ RearrangeB(G, C_G) ▷ rearranges according to the increasing order of C_G values
$G \leftarrow$ Prune(G, K)
loop:
if $k < H$ **then**
    $G \leftarrow$ GenMeas(G) ▷ generates the sequence of covariance matrices at level k given matrices from level $k-1$
    $C_G \leftarrow$ WgtAvg(G)
    $G \leftarrow$ RearrangeB(G, C_G)
    $G \leftarrow$ Prune(G, K)
$i \leftarrow i + 1$.
goto *loop*.

After implementing either of the pruning methods described above, we discard all the branches in the tree that end prematurely, i.e., branches of length less than H. With these pruning algorithms, the pre-computation complexity and the number of binary decision variables grow only polynomially with respect to the length of the planning horizon H. With pruning, the above MINLP boils down to the following.

Let $v_{ij}, i = 1, \ldots, H, j = 1, \ldots, p_i$ be a collection of new binary variables, where p_i is the number of surviving nodes at time-step i after pruning. Since we are discarding branches (along with the nodes in the branches) that end prematurely, $p_i \leq K\ \forall i$. Let *Child(v_{ij})* represent the set of binary variables corresponding to the child nodes of the node v_{ij}, which is predetermined $\forall i, j$. Therefore, the UAV path optimization problem becomes

$$\underset{v_{ij};\ j=1,\ldots,p_i\ \ i=1,\ldots,H}{\text{minimize}} \quad \sum_{i=1}^{H} \text{trace}(A_i^{-1})$$

$$\text{s.t.} \quad A_i = P_{i|i-1}^{-1} + H^T \left(\sum_{j=1}^{p_i} R_j^{-1} v_{ij} \right) H;\ \forall i = 1, \ldots, H$$

$$P_{i|i-1} = F A_{i-1}^{-1} F^T + Q,\ \forall i = 1, \ldots, H$$

$$\sum \textit{Child}(v_{ij}) = v_{ij},\ \forall j, i = 1, \ldots, H-1$$

$$\sum_j v_{1j} = 1$$

$$v_{ij} = \{0, 1\}, \quad \forall i, j.$$

We implement a scenario with a single UAV tracking a single target to test the performance of the above-mentioned methods. We plan the controls for the UAV (lateral accelerations and bank angles) over a horizon of length $H = 5$ by solving the above MINLP problems. For further simplification, we optimize only the bank angles for the UAV and keep the lateral accelerations to zero (i.e., we assume fixed-speed UAVs), and specifically the bank angles (controls) are chosen from the discrete set $\{-5, -3, -1, 1, 3, 5\}$. With these assumptions, we first solve the MINLP with no pruning, and then with pruning with $K = 10, 100, 300, 800$. The pruning size $K = 10$ means that at any level in the tree (shown in Figure 6), we allow at most only 10 nodes and the rest are pruned. Figure 7 shows the final objective function values for the methods discussed above for various pruning lengths. We notice that the increase in pruning length increases the performance of MINLP with PrunMethod-B. However, with PrunMethod-A, our simulations suggest that increasing the pruning length may not guarantee increase in the performance of the MINLP, as is evident from Figure 7. The reason PrunMethod-B is well behaved is that the pruning criteria is based on the historical performance of the branch (in the tree model described earlier) along with the performance of the nodes at the current level, unlike PrunMethod-A, which is a greedy pruning strategy that takes into account only the performance of the nodes at the current level. As expected, Figure 8 shows that the execution time can be greatly reduced with pruning, but at the cost of losing optimality (increased final objective function value).

Our numerical results confirm that MATLAB's *fmincon* still outperforms the above described MINLP formulation. The reason is that the MINLP formulation, although capable of finding the global optimum solution, is losing the optimality in the control space discretization step as we kept the

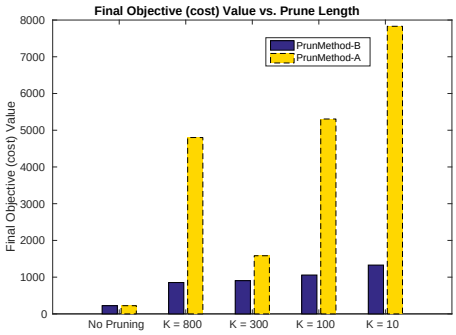

Fig. 7. Final objective function values: MINLP Methods with and without Pruning

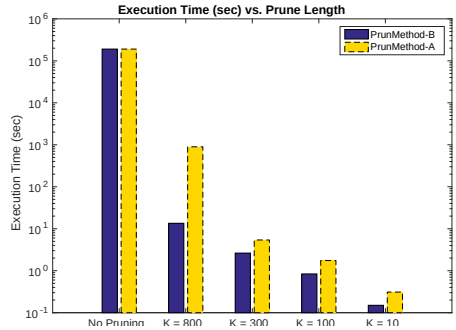

Fig. 8. Execution time: MINLP Methods with and without Pruning

resolution of the discretization low to keep the computational intensity low. Thus, by allowing for additional computational intensity, we can find the global optimum solution (if not, at least outperform local solvers) with our MINLP formulation by increasing the resolution of the control space discretization. It is clear that the MPC type framework does not readily allow the discretization approaches of the form successfully applied in [16]. But, the effort to find suboptimal solutions is also worthwhile as finding the exact solution for high-dimensional non-convex problems (like the one presented in this paper) can be computationally too expensive in practice.

IV. Conclusions

We studied the performance of an MINLP formulation in the context of UAV path optimization problems. Without loss of generality, we chose a target tracking application. Our numerical results suggest that the *knitro* solver (with and without multi-start) was able to outperform MATLAB's *fmincon* solver. Although the solver *knitro (with multi-start)* may seem computationally more demanding compared to *fmincon*, *knitro* has a parallelization feature, which when utilized makes *knitro*'s computational requirement almost the same as *fmincon*.

We then formulated the problem as an MINLP in an attempt to find the global optimum solution, and presented two *pruning* methods to reduce the computational complexity of the MINLP. We presented a numerical study, and the results clearly showed that the *pruning* methods reduce the computation time significantly, however, at the cost of the final objective value drifting away from the optimum.

Our initial work on MINLP formulation (presented in this study) does not yet show that it can outperform the local solver *fmincon*. The reason for this under-performance is that the MINLP formulation is losing the optimality in the control space discretization step as we kept the resolution low in order to reduce the computational effort. Thus, if we can tolerate the additional computational intensity, we can find the global optimum (if not, at least outperform local solvers) with our MINLP formulation by increasing the resolution of the control space discretization.

References

[1] S. A. P. Quintero, D. A. Copp, and J. P. Hespanha, "Robust UAV coordination for target tracking using output-feedback model predictive control with moving horizon estimation," in *Proc. American Control Conf.*, Chicago, IL, 2015, pp. 3758–3764.

[2] D. van der Walle, B. Fidan, A. Sutton, C. Yu, and B. D. O. Anderson, "Non-hierarchical UAV formation control for surveillance tasks," in *Proc. American Control Conf.*, Seattle, WA, June 2008, pp. 777–782.

[3] I. Shames, B. Fidan, and B. D. O. Anderson, "Close target reconnaissance using autonomous UAV formations," in *Proc. 47th IEEE Conf. Decision and Control*, Cancun, Mexico, Dec. 2008, pp. 1729–1734.

[4] S. Ragi and E. K. P. Chong, "UAV path planning in a dynamic environment via partially observable Markov decision process," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, pp. 2397–2412, 2013.

[5] ——, "Dynamic UAV path planning for multitarget tracking," in *Proc. American Control Conf.*, Montreal, Canada, 2012, pp. 3845–3850.

[6] ——, "Decentralized guidance control of UAVs with explicit optimization of communication," *J. Intelligent & Robotic Systems*, vol. 73, no. 1, pp. 811–822, 2014.

[7] E. J. Forsmo, E. I. Grtli, T. I. Fossen, and T. A. Johansen, "Optimal search mission with unmanned aerial vehicles using mixed integer linear programming," in *Proc. 2013 Int. Conf. Unmanned Aircraft Systems*, Atlanta, GA, 2013, pp. 253–259.

[8] V. M. Goncalves, L. C. A. Pimenta, C. A. Maia, and G. A. S. Pereira, "Optimal search mission with unmanned aerial vehicles using mixed integer linear programming," in *Proc. 2013 IEEE Int. Conf. Robotics and Automation*, Karisruhe, Germany, 2013, pp. 849–854.

[9] S. A. Miller, Z. A. Harris, and E. K. P. Chong, "A POMDP framework for coordinated guidance of autonomous UAVs for multitarget tracking," *EURASIP J. Adv. Signal Process.*, vol. 2009, 2009.

[10] J. S. Bellingham, M. Tillerson, M. Alighanbari, and J. P. How, "Cooperative path planning for multiple UAVs in dynamic and uncertain environments," in *Proc. 41st IEEE Conf. Decision and Control*, Las Vegas, Nevada, Dec. 2002, pp. 2816–2822.

[11] M. Zhang and D. Kingston, "Time-space network based exact models for periodical monitoring routing problem," in *Proc. American Control Conf.*, Chicago, IL, July 2015, pp. 5264–5269.

[12] E. I. Grøtli and T. A. Johansen, "Task assignment for cooperating UAVs under radio propagation path loss constraints," in *Proc. American Control Conf.*, Montreal, Candada, June 2012, pp. 3278–3283.

[13] F. Borrelli, D. Subramanian, A. U. Raghunathan, and L. T. Biegler, "Milp and nlp techniques for centralized trajectory planning of multiple unmanned air vehicles," in *Proc. American Control Conf.*, Minneapolis, MN, June 2006, pp. 5763–5768.

[14] A. J. Eele and A. Richards, "Path-planning with avoidance using nonlinear branch-and-bound optimization," *J. Guid. Control Dynam.*, vol. 32, pp. 384–394, Mar. 2009.

[15] C. Reinl and O. V. Stryk, "Optimal control of multi-vehicle-systems under communication constraints using mixed-integer linear programming," in *Proc. 1st Int. Conf. Robot Communication and Coordination*, Athens, Greece, 2007, pp. 1–8.

[16] H. D. Mittelmann and D. Salvagnin, "Exact and heuristic approaches for directional sensor control," *IEEE Sensors J.*, vol. 15, no. 11, pp. 6633–6639, 2015.

[17] R. Oberdieck and E. N. Pistikopoulos, "Explicit hybrid model-predictive control: The exact solution," *Automatica*, vol. 58, pp. 152–159, 2015.

[18] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 2, pp. 267–278, 2010.