An External Active-Set Strategy for Solving Optimal Control Problems

Hoam Chung, *Member, IEEE*, Elijah Polak, *Life Fellow, IEEE*, and Shankar Sastry, *Fellow, IEEE*

***Abstract*—We present a new, *external*, active constraints set strategy for solving nonlinear programming problems with a large number of inequality constraints that arise in the process of discretizing continuous-time optimal control problems with state-space constraints. This strategy constructs a sequence of inequality constrained nonlinear programming problems, containing a progressively larger subset of the constraints in the original problem, and submits them to a nonlinear programming solver for a fixed number of iterations. We prove that this scheme computes a solution of the original problem and show by means of numerical experiments that this strategy results in reductions in computing time ranging from a factor of 6 to a factor of over 100.**

***Index Terms*—Active-set strategies, nonlinear programming, optimal control.**

I. Introduction

Optimal control problems with state space constraints are usually solved by discretizing the dynamics, which results in the conversion of the continuous-time optimal control problem into a nonlinear programming problem. This conversion can be done in one of two ways. The first is called *collocation*, see, e.g. [1], which treats the control and the state as independent variable. It results in nonlinear programming problems with a very large number of variables, a very large number of nonlinear (collocation) equality constraints, representing the dynamics, and a large number of nonlinear inequality constraints, representing the state space constraints. Its main advantage is that the expressions for derivatives are relatively simple. The second way is to replace the continuous optimal control problem by a discrete-time optimal control problem, obtained by integrating the dynamics using Euler's method. Some Runge-Kutta methods can also be used, but they lead to considerable complications (see, [2]). A discrete-time optimal control problem is also a nonlinear programming problem. Its distinguishing features are (i) only the control is a decision variable, and hence the number of design variables is relatively small, (ii) there are no nonlinear equality constraints representing the dynamics, since the state is computed explicitly, (iii) assuming the same discretization grid-size of time, the number of state-space inequality constraints is the same as in collocation, and (iv) the required gradients can be computed using adjoint equations.

With either method, although the number of state-space inequality constraints may be large, relatively few of these inequalities are active. The active-set strategy that we will discuss in this technical note can be used in conjunction with either transcription into a nonlinear programming problem. However, we have a preference for the transcription into a discrete-time optimal control problem, for the following reasons, see [3], Chapter 4. First, the discrete-time optimal control problems are consistent approximations to the continuous-time problems, and hence cannot have spurious local minima or stationary points.[1] Second, they are compatible with adaptive discretization schemes, which result in considerable computing time savings. And third, it is very simple to scale the finite-dimensional control and initial state variables so as to preserve the norms they have in the continuous-time case. When this scaling is not preserved, serious ill-conditioning can result.

An important example of optimal control problems with state space constraints arises in the trajectory planning for unmanned aerial vehicles (UAV's) using receding horizon control (RHC). RHC is a form of sample-data control that determines the control to be applied over the next sampling interval by solving an optimal control problem during the current sampling interval. The optimal control problems for RHC control of UAV's are characterized by large numbers of collision avoidance inequalities and by expensive evaluations of the gradients of these inequality defining functions. Since potential collisions are confined to relatively short segments of the UAV trajectories, most of the collision avoidance inequalities are inactive. In the case of UAV's, the sampling intervals are short and hence the viability of the RHC scheme is largely determined by the speed of the nonlinear programming solvers.

Unfortunately, standard nonlinear programming packages, including the excellent set found in TOMLAB [4], including SNOPT [5], NPSOL [6], Schittkowski SQP [7], and KNITRO[2] [8], are not designed to exploit the fact that a problem with a large number of nonlinear inequality constraints may have few active constraints. A scan of the literature reveals a few algorithms with built-in active set strategies, such as the first-order Phase I-Phase II methods of feasible directions (see [3]), the superlinearly converging CFSQP [9], and the minimax algorithm [10]. However, these active-set strategies are not transferable to existing packages, such as those mentioned earlier. There appear to be no "external" active-set strategies in the literature that can be added via a small subroutine to an existing nonlinear programming package.

In this technical note, we present what appears to be the first *external* active constraint set strategy for solving nonlinear programming problems with a large number of inequality constraints. This strategy constructs a sequence of inequality constrained nonlinear programming problems, containing a progressively larger subset of the constraints in the original problem, and submits them to a nonlinear programming solver for a fixed number of iterations. We prove that this scheme computes a solution of the original problem and show by means of numerical experiments that when applied to UAV trajectory planning, this strategy results in reductions in computing time ranging from a factor of 6 to a factor of over 100. Our new strategy is particularly effective when used with nonlinear programming solvers that allow warm starts. It may be useful to observe that a related strategy [10] for solving semi-infinite minimax problems using log-sum-exponential smoothing has proved to be equally effective.

In Section II we state our strategy in the form of an algorithm and provide theoretical justification for it, in Section III we present numerical results, and our concluding remarks are in Section IV.

Notation: We will denote elements of a vector by superscripts (e.g., x^i) and elements of a sequence or a set by subscripts (e.g., η_k). ∎

[1]Points satisfying first-order conditions of optimality.

[2]KNITRO is a collection of optimization algorithms, and we use the algorithm option "Interior/Direct" with quasi-Newton symmetric rank one updates in this paper.

Manuscript received April 02, 2008; revised October 19, 2008. Current version published May 13, 2009. This work was supported by ARO MURI SWARMS (W911NF-0510219) and ARO Phase II STTR (W911NF-06-C-0192). Recommended by Associate Editor F. Bullo.

The authors are with the Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720 USA (e-mail: hachung@eecs.berkeley.edu; polak@eecs.berkeley.edu; sastry@eecs.berkeley.edu).

Color versions of one or more of the figures in this technical note are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2009.2013035

II. The Algorithm

Consider the inequality constrained minimization problem:

$$\mathbf{P_q} \quad \min\left\{f^0(\eta) | f^j(\eta) \leq 0, j \in \mathbf{q}\right\} \tag{1}$$

where $\eta \in \mathbb{R}^n$, and $\mathbf{q} \triangleq \{1,\ldots,q\}$. We assume that functions $f^j : \mathbb{R}^n \to \mathbb{R}$ are at least once continuously differentiable. In the context of discrete optimal control problems, η can be either a discretized control sequence or an initial state - discretized control sequence pair.[3]

Next we define the index set $\mathbf{q}_\epsilon(\eta)$ with $\epsilon > 0$ by

$$\mathbf{q}_\epsilon(\eta) \triangleq \left\{j \in \mathbf{q} | f^j(\eta) \geq \psi_+(\eta) - \epsilon\right\} \tag{2}$$

where

$$\psi(\eta) \triangleq \max_{j \in \mathbf{q}} f^j(\eta) \tag{3}$$

and

$$\psi_+(\eta) \triangleq \max\left\{0, \psi(\eta)\right\}. \tag{4}$$

Referring to any advance textbook on nonlinear programming (see, e.g., [3], [11], [12]), we see that the convergence of algorithms such as Phase I- Phase II Methods of Feasible Directions, various versions of Sequential Quadratic Programming, Augmented Lagrangian methods, etc, in solving problems of the form (1), depend on these problems satisfying a number of assumptions, such as that the functions are once or twice continuously differentiable, that there is an interior feasible point (Slater' constraint qualification), and that the problems have optimal solutions,[4] as well as some other more algorithm specific condition. Implicitly, we assume that the problems that we wish to solve and the algorithms that we use are appropriately matched. Hence we assume that we will be using convergent algorithms, defined as follows.

Definition: We say that an algorithm defined by a recursion of the form

$$\eta_{k+1} = A(\eta_k) \tag{5}$$

for solving inequality constrained problems of the form (1), is *convergent* if any accumulation point[5] of a sequence $\{\eta_i\}_{i=0}^{\infty}$, constructed according to the recursion (5), is a feasible stationary point.[6] ■

To complete the elements for our scheme, we assume that the inequality constrained problems of interest, in the form (1), satisfy appropriate assumptions and that we have a convergent algorithm for their solution. In fact, in our numerical experiments, we use several convergent algorithms.

Algorithm 1: Active-Set Algorithm for Inequality Constrained Minimization Problems

```
Data: $\eta_0$, $\epsilon > 0$, $N_{iter} \in \mathbb{N}$
Step 0: Set $i = 0$, $\mathbf{Q}_0 = \mathbf{q}_\epsilon(\eta_0)$.
Step 1: Set $\zeta_0 = \eta_i$ and perform $N_{iter}$ iterations of the form
$\zeta_{k+1} = A(\zeta_k)$ on the problem
        $\mathbf{P}_{\mathbf{Q}_i} \quad \min\left\{f^0(\zeta) | f^j(\zeta) \leq 0, \quad j \in \mathbf{Q}_i\right\}$   (6)
to obtain $\zeta_{N_{iter}}$ and set $\eta_{i+1} = \zeta_{N_{iter}}$.
Step 2: Compute $\psi(\eta_{i+1})$.
if $\zeta_{N_{iter}}$ is returned as a global, local, or stationary solution of $\mathbf{P}_{\mathbf{Q}_i}$
and $\psi(\eta_{i+1}) \leq 0$, then
    STOP,
else
    Compute
        $\mathbf{Q}_{i+1} = \mathbf{Q}_i \cup \mathbf{q}_\epsilon(\eta_{i+1})$   (7)
    and set $i = i + 1$, and go to Step 1.
end if
```

Lemma 2: Suppose that $\epsilon > 0$ and that the sequence $\{\eta_i\}_{i=0}^{\infty}$, in $\mathbb{R}^n$, is such that $\eta_i \to \hat{\eta}$ as $i \to \infty$. Then there exists an i_0 such that for all $i \geq i_0$, $\mathbf{q}_0(\hat{\eta}) \subseteq \mathbf{q}_\epsilon(\eta_i)$.

Proof: By definition (2), for any $j \in \mathbf{q}_0(\hat{\eta})$,

$$f^j(\hat{\eta}) - \psi_+(\hat{\eta}) \leq 0. \tag{8}$$

First suppose that $\psi_+(\hat{\eta}) = \psi(\hat{\eta}) \geq 0$. Then the set $\mathbf{q}_0(\hat{\eta})$ is nonempty and $f^j(\hat{\eta}) - \psi_+(\hat{\eta}) = 0$, for all $j \in \mathbf{q}_0(\hat{\eta})$. Since all the functions $f^j(\cdot)$ and $\psi(\cdot)$ are continuous, for all $j \in \mathbf{q}_0(\hat{\eta})$, $[f^j(\eta_i) - \psi_+(\eta_i)] \to [f^j(\hat{\eta}) - \psi_+(\hat{\eta})] = 0$ as $i \to \infty$. Hence there must exist an i_0 such that for all $i \geq i_0$ and $j \in \mathbf{q}_0(\hat{\eta})$,

$$f^j(\eta_i) - \psi_+(\eta_i) \geq -\epsilon \tag{9}$$

which proves that for all $i \geq i_0$, $\mathbf{q}_0(\hat{\eta}) \subseteq \mathbf{q}_\epsilon(\eta_i)$.

Next suppose that $\psi(\hat{\eta}) < 0$. Then $\psi_+(\hat{\eta}) = 0$, and the set $\mathbf{q}_0(\hat{\eta})$ is empty. Since the empty set is a subset of any set, the desired result follows. ■

Lemma 3: Suppose that $\epsilon > 0$ and that the sequence $\{\eta_i\}_{i=0}^{\infty}$, in $\mathbb{R}^n$, is such that $\eta_i \to \hat{\eta}$ as $i \to \infty$ and that $\mathbf{Q} = \cup_{i=0}^{\infty} \mathbf{q}_\epsilon(\eta_i)$. For any $\eta \in \mathbb{R}^n$, let

$$\psi_{\mathbf{Q}}(\eta) = \max_{j \in \mathbf{Q}} f^j(\eta). \tag{10}$$

If $\psi_{\mathbf{Q}}(\hat{\eta}) \leq 0$, then $\psi(\hat{\eta}) \leq 0$.

Proof: By Lemma 2, there exists an i_0 such that $\mathbf{q}_0(\eta_i) \subseteq \mathbf{Q}$ for all $i \geq i_0$, and hence it follows that $\psi(\eta_i) = \psi_{\mathbf{Q}}(\eta_i)$ for all $i \geq i_0$. Now, both $\psi(\cdot)$ and $\psi_{\mathbf{Q}}(\cdot)$ are continuous, and hence $\psi(\hat{\eta}) = \lim_{i\to\infty} \psi(\eta_i) = \lim_{i\to\infty} \psi_{\mathbf{Q}}(\eta_i) = \psi_{\mathbf{Q}}(\hat{\eta})$. The desired result now follows directly. ■

Lemma 4: Suppose that $\mathbf{Q} \subseteq \mathbf{q}$ and consider the problem

$$\mathbf{P_Q} \quad \min\left\{f^0(\eta) | f^j(\eta) \leq 0, j \in \mathbf{Q}\right\}. \tag{11}$$

Suppose that $\hat{\eta} \in \mathbb{R}^n$ is feasible for $\mathbf{P_q}$, i.e, $f^j(\eta) \leq 0$ for all $j \in \mathbf{q}$.

a) If $\hat{\eta}$ is a global minimizer for $\mathbf{P_Q}$, then it is also a global minimizer for $\mathbf{P_q}$.

b) If $\hat{\eta}$ is a local minimizer for $\mathbf{P_Q}$, then it is also a local minimizer for $\mathbf{P_q}$.

c) If $\hat{\eta}$ is a stationary point for $\mathbf{P_Q}$, then it is also a stationary point for $\mathbf{P_q}$.

Proof: Clearly, since $\hat{\eta}$ is feasible for $\mathbf{P_q}$ it is also feasible for $\mathbf{P_Q}$.

[3]see [3] Chapter 4 for a detailed treatment of discrete time optimal control problems.

[4]The problem $min_x e^{-x}$ does not have an optimal solution.

[5]A point $\hat{\eta}$ is said to be an accumulation point of the sequence $\{\eta_i\}_{i=0}^{\infty}$, if there exists an infinite subsequence, indexed by $K \subseteq \mathbb{N}$, $\{\eta_i\}_{i\in K}$, such that $\eta_i \xrightarrow{K} \hat{\eta}$. as $i \xrightarrow{K} \infty$.

[6]A point η is called as a stationary point or a stationary solution of $\mathbf{P}_{\mathbf{Q}_i}$ and $\psi(\eta_{i+1}) \leq 0$ if it satisfies the F. John conditions [13] (or Theorem 2.2.4, p. 188 in [3]).

a) Suppose that $\hat{\eta}$ is not a global minimizer for $\mathbf{P_q}$. Then there exists an η^* such that $f^j(\eta^*) \leq 0$ for all $j \in \mathbf{q}$ and $f^0(\eta^*) < f^0(\hat{\eta})$. Now, η^* is also feasible for $\mathbf{P_Q}$ and hence $\hat{\eta}$ cannot be a global minimizer for $\mathbf{P_Q}$, a contradiction.

b) Suppose that $\hat{\eta}$ is not a local minimizer for $\mathbf{P_q}$. Then there exists a sequence $\{\eta_i\}_{i=0}^{\infty}$ such that $\eta_i \to \hat{\eta}$, $f^0(\eta_i) < f^0(\hat{\eta})$ and $f^j(\eta_i) \leq 0$ for all i and $j \in \mathbf{q}$. But this contradicts the assumption that $\hat{\eta}$ is a local minimizer for $\mathbf{P_Q}$.

c) Since $\hat{\eta}$ satisfies the F. John conditions for $\mathbf{P_Q}$, there exist multipliers $\mu^0 \geq 0$, $\mu^j \geq 0$, $j \in \mathbf{Q}$, such that $\mu^0 + \sum_{j\in\mathbf{Q}} \mu^j = 1$,

$$\mu^0 \nabla f^0(\hat{\eta}) + \sum_{j\in\mathbf{Q}} \mu^j \nabla f^j(\hat{\eta}) = 0 \tag{12}$$

and

$$\sum_{j\in\mathbf{Q}} \mu^j f^j(\hat{\eta}) = 0. \tag{13}$$

Clearly, $\hat{\eta}$ also satisfies the F. John conditions for $\mathbf{P_q}$ with multipliers $\mu^j = 0$ for all $j \notin \mathbf{Q}$ and otherwise as for $\mathbf{P_Q}$. ∎

Combining the above lemmas, we get the following convergence result.

Theorem 5: Suppose that the problem $\mathbf{P_q}$ has feasible solutions, i.e., there exist vectors η^* such that $f^j(\eta^*) \leq 0$ for all $j \in \mathbf{q}$.

a) If Algorithm 1 constructs a finite sequence $\{\eta_i\}_{i=0}^{k}$, exiting in Step 2, with $i+1 = k$, then η_k is a global, local, or stationary solution for $\mathbf{P_q}$, depending on the exit message from the solver defined by $A(\cdot)$.

b) If $\{\eta_i\}_{i=0}^{\infty}$ is an infinite sequence constructed by Algorithm 1 in solving $\mathbf{P_q}$, then any accumulation point $\hat{\eta}$ of this sequence is feasible and stationary for $\mathbf{P_q}$.

Proof:

a) If the sequence $\{\eta_i\}_{i=0}^{k}$ is finite, then, by the exit rule, it is feasible for $\mathbf{P_q}$ and it is a global, local, or stationary solution for $\mathbf{P}_{\mathbf{Q}_i}$. It now follows from Lemma 4, that it is also a global, local, or stationary solution for $\mathbf{P_q}$.

b) Since the sets $\mathbf{Q}_i$ grow monotonically, and since $\mathbf{q}$ is finite, there must exist an i_0 and a set $\mathbf{Q} \subseteq \mathbf{q}$, such that $\mathbf{Q}_i = \mathbf{Q}$ for all $i \geq i_0$. Next, it follows from the fact that $A(\cdot)$ is convergent, that for any accumulation point $\hat{\eta}$, $\psi_{\mathbf{Q}}(\hat{\eta}) \leq 0$. Let K be an infinite subset of the positive integers, such that the subsequence $\{\eta_i\}_{i\in K}$ converges to $\hat{\eta}$. Since the accumulation point $\hat{\eta}$ is the limit point of the subsequence $\{\eta_i\}_{i\in K}$, it follows from Lemma 2 that $\mathbf{q}_0(\hat{\eta}) \subseteq \mathbf{Q}$. Applying Lemma 3 to this subsequence, we conclude that $\psi(\hat{\eta}) \leq 0$, i.e., that $\hat{\eta}$ is a feasible point for $\mathbf{P_q}$. It now follows from the fact that $A(\cdot)$ is convergent and Lemma 4 that any accumulation point $\hat{\eta}$ is stationary for $\mathbf{P_q}$. ∎

III. Numerical Results

All numerical experiments were performed using MATLAB V7.2 and TOMLAB V5.5 [4] in Windows XP, on a desktop computer with an Intel Xeon 3.2 GHz processor with 3 GB RAM. Optimization solvers tested in this technical note were the Schittkowski SQP algorithm with cubic line search [7], NPSOL 5.02 [6], SNOPT 6.2 [5], and KNITRO [8]. It should be clear from the form of Algorithm 1, that our strategy benefits considerably from warm starts of the nonlinear programming solvers, to be used after the construction of the active set $\mathbf{Q}_i$. Hence it is desirable to use solvers with as extensive a warm start capability as possible, so that one can transmit the last value of important information from the last iteration of a solver on the problem $\mathbf{P}_{\mathbf{Q}_i}$ as initial conditions for solving the problem $\mathbf{P}_{\mathbf{Q}_{i+1}}$. SNOPT allows the user to provide initial variables and their states and slack variables. NPSOL allows the user to provide initial variables and their states, Lagrange multipliers, as well as an initial Hessian approximation matrix for quasi-Newton updates. conSolve, the TOMLAB implementation of the Schittkowski SQP algorithm, allows the user to provide an initial solution vector and initial Hessian matrix approximation. KNITRO allows the user to provide only the initial solution vector. For maximum efficiency, this data must be saved at the end of the i-th run and transmitted as initial data for the $i+1$-th run of the solver.

As we will see from our numerical results, the performance of Algorithm 1 is much more sensitive to the parameters N_{iter} and ϵ when using a solver, such as KNITRO, that has minimal warm start features, than when it is using a solver with extensive warm start features.

A. Trajectory Planning for Multiple Fixed-Wing UAVs

Our numerical example consists of controlling multiple UAVs. Suppose that we have N_a UAVs, each with the same dynamics. We assume that each UAV flies at a constant speed v and that the scalar control u determines the yaw rate of the UAV. In spite of its simplicity, this model describes quite well the dynamics of a fixed-wing aircraft, whose attitude, altitude, and forward speed are stabilized by an autopilot. It was used in other papers, including air traffic management [14], and UAV trajectory planning [15]. In order to state the optimal control problem as an end-point problem defined on [0, 1], we rescale the state dynamics using the actual terminal time T, x^4, and augment the 3-dimensional physical state with a fourth component, so that

$$x^4(t) = \int_0^t \frac{T}{2} u^2(\tau) d\tau \tag{14}$$

represents the energy used. The resulting dynamics have the form

$$\frac{d\mathbf{x}^i}{dt} = \frac{d}{dt}\begin{bmatrix} x^{1,i} \\ x^{2,i} \\ x^{3,i} \\ x^{4,i} \end{bmatrix} = \begin{bmatrix} Tv\cos x^{3,i} \\ Tv\sin x^{3,i} \\ Tu^i \\ \frac{T}{2}(u^i)^2 \end{bmatrix} \triangleq h\left(\mathbf{x}^i(t), u^i(t)\right) \tag{15}$$

where $i \in \{1, 2, \ldots, N_a\}$ denotes the UAV index, with the initial state $\mathbf{x}^i(0)$ given for all i. Here, the components of the state $(x^{1,i}, x^{2,i})$ represent the position of the i-th UAV in the plane, and the component $x^{3,i}$ represents the heading. We will denote the solution of the dynamic (15) by $\mathbf{x}^i(t, u^i)$, with $t \in [0, 1]$.

We want them to stay in a circular region centered at the origin, without incurring any collisions. The stay-in-a-circle constraints are described by the following equations:

$$f_{\text{bnd}}^{t,i}(u^i) \triangleq x^{1,i}(t, u^i)^2 + x^{2,i}(t, u^i)^2 \leq r_{\text{bnd}}^2, \quad t \in [0, 1]. \tag{16}$$

The collision avoidance constraints are given by

$$\begin{aligned} f_{\text{ca}}^{t,(i,j)}(u^i, u^j) \triangleq & \left(x^{1,i}(t, u^i) - x^{1,j}(t, u^j)\right)^2 \\ & + \left(x^{2,i}(t, u^i) - x^{2,j}(t, u^j)\right)^2 \\ & \geq r_{\text{ca}}^2, \quad t \in [0, 1], \end{aligned} \tag{17}$$

where (i, j) is an element of the set of all 2-combinations of the index set $\{1, 2, \ldots, N_a\}$.

The optimal control problem has the form

$$\min_{u^i \in \mathbb{R}, i \in \{1,\ldots,N_a\}} f^0(u) \triangleq \sum_{i=1}^{N_a} x^{4,i}(1, u^i) \tag{18}$$

subject to stay-in-a-circle constraints (16) and collision avoidance constraints (17).

In order to solve this problem, we must discretize the dynamics. We use Euler's method to obtain the discretized dynamics

$$\bar{\mathbf{x}}(t_{k+1}) - \bar{\mathbf{x}}(t_k) = \Delta h\left(\bar{\mathbf{x}}(t_k), \bar{\mathbf{u}}(t_k)\right), \ \ \bar{x}(0) = x(0) \tag{19}$$

with $\Delta \triangleq 1/N$, $N \in \mathbb{N}$, $t_k \triangleq k\Delta$ and $k \in \{0, 1, \ldots, N\}$. We use an overbar to distinguish between the exact variables and the discretized variables. We will denote the solution of the discretized dynamics by $\bar{\mathbf{x}}^i(t_k, \bar{\mathbf{u}}^i)$, $k = 0, 1, \ldots, N$. Letting

$$\bar{\mathbf{u}}^i \triangleq \left(\bar{\mathbf{u}}^i(t_0), \bar{\mathbf{u}}^i(t_1), \ldots, \bar{\mathbf{u}}^i(t_{N-1})\right), \text{ and } \bar{\mathbf{u}} \triangleq \begin{bmatrix} \bar{\mathbf{u}}^1 \\ \vdots \\ \bar{\mathbf{u}}^{N_a} \end{bmatrix} \tag{20}$$

we obtain the discretized optimal control problem

$$\min_{\bar{\mathbf{u}}^i \in \mathbb{R}^N, i \in \{1,\ldots,N_a\}} \bar{f}^0(\bar{\mathbf{u}}) \triangleq \sum_{i=1}^{N_a} \bar{x}^{4,i}(1, \bar{\mathbf{u}}^i) \tag{21}$$

subject to the constraints

$$\bar{f}^{k,i}_{\text{bnd}}(\bar{\mathbf{u}}^i) \triangleq \bar{x}^{1,i}(t, \bar{\mathbf{u}}^i)^2 + \bar{x}^{2,i}(t, \bar{\mathbf{u}}^i)^2 \leq r^2_{\text{bnd}}, \ \ k \in \{1, \ldots, N\} \tag{22}$$

and

$$\begin{aligned} \bar{f}^{k,(i,j)}_{\text{ca}}(\bar{\mathbf{u}}^i, \bar{\mathbf{u}}^j) \triangleq & \left(\bar{x}^{1,i}(k, \bar{\mathbf{u}}^i) - \bar{x}^{1,j}(k, \bar{\mathbf{u}}^j)\right)^2 \\ & + \left(\bar{x}^{2,i}(k, \bar{\mathbf{u}}^i) - \bar{x}^{2,j}(k, \bar{\mathbf{u}}^j)\right)^2 \\ & \geq r^2_{\text{ca}}, \ \ k \in \{1, \ldots, N\}. \end{aligned} \tag{23}$$

The total number of inequality constraints in this problem is $N\ N_a(N_a - 1)/2 + N\ N_a$.

Clearly, (21) is a mathematical programming problem which is distinguished from ordinary mathematical programming problems only by the fact that adjoint equations can be used in the computation of the gradients of the functions $f^k(\cdot)$ and $f^{k,(i,j)}$. Another special feature of this problem is that it has a very large number of inequality constraints, of which only a small number are likely to be active due to the nature of the dynamics.

The dynamics of the UAV's are nonlinear and the non-collision constraints are non-convex. Hence this problem has many local minima. Consequently, the solution trajectory may depend on the initial control and on the evolution of the active sets during computation.

B. Numerical Results

For numerical experiments, we set $r_{\text{bnd}} = 4$, $r_{\text{ca}} = 1$, $N = 64$ and $N_a = 4$, resulting in 640 nonlinear inequality constraints. The initial conditions and initial controls for each agent are set as

$$\begin{aligned} x^1_0 &= (2.5, 2.5, \pi, 0), \quad x^2_0 = (-2.5, 2, -\pi/2, 0) \\ x^3_0 &= (-2.5, -2.5, -\pi/4, 0), \quad x^4_0 = (2, -2.5, \pi/2, 0) \\ \bar{u}^i_0 &= 1.25 \times 10^{-1} \mathbf{1}_{1\times N}, \ i \in \{1, 2, 3, 4\}. \end{aligned} \tag{24}$$

The numerical results are summarized in the Tables I–IV. In these tables, N_{grad}, the total number of gradient evaluations, and t_C, the total CPU time for achieving an optimal solution using Algorithm 1, are defined as follows:

$$\begin{aligned} N_{grad} = & \sum_{i=0}^{i_T} |\mathbf{Q}_i| \times (\text{number of gradient function calls} \\ & \text{during } i-\text{th inner iteration}) \\ t_C = & \sum_{i=0}^{i_T} [\text{CPU time spent for } i-\text{th inner iteration} \\ & + \text{CPU time spent for setting up} \\ & i-\text{th inner iteration}]. \end{aligned} \tag{25}$$

In the above, and in the tables, i_T is the value of the iteration index i at which Algorithm 1 is terminated by the termination tests incorporated in the optimization solver used, and i_{stab} is the value of index i at which $|\mathbf{Q}_i|$ is stabilized. Also, $\%_R$, the percentage of t_C with respect to the computation time with the *raw* algorithm, i.e. using the solver with the full set of constraints (shown in the last row of each table), is used in all tables.

Fig. 1 shows a locally optimal solution for this problem. There are only 8 active constraints at the end. These are all associated with staying in the circle; there are no active non-collision constraints. With certain values of ϵ, Algorithm 1 accumulates fewer than 16 constraints. Consequently, the reduction in the number of gradient computations is huge, and the savings in CPU time is more dramatic than in the single UAV case. There exist parameter pairs (ϵ and N_{iter}) which, when used with conSolve and KNITRO, achieve over 95% savings in computation

TABLE I
EXTERNAL ACTIVE-SET STRATEGY WITH SCHITTKOWSKI SQP ALGORITHM, FOUR-UAV EXAMPLE

#	ϵ	N_{iter}	i_T	f^0	N_{grad}	$\lvert\mathbf{Q}\rvert$	t_C	$\%_R$
01	1	10	17	1.379	27623	122	292	15.2
02	1	20	16	1.071	41556	154	422	22.0
03	1	30	7	1.159	16626	77	180	9.4
04	0.1	10	14	1.420	6444	44	90.9	4.7
05	0.1	20	11	0.803	9463	30	129	6.7
06	0.1	30	12	1.097	16393	32	229	11.9
07	0.01	10	19	1.071	4477	28	83.2	4.3
08	0.01	20	16	1.071	6318	26	115	6.0
09	0.01	30	11	0.803	5810	16	109	5.7
Raw				3.356	274560	640	1918	100

TABLE II
EXTERNAL ACTIVE-SET STRATEGY WITH NPSOL, FOUR-UAV EXAMPLE

#	ϵ	N_{iter}	i_T	f^0	N_{grad}	$\lvert\mathbf{Q}\rvert$	t_C	$\%_R$
01	1	10	12	1.108	12384	109	133	1.9
02	1	20	10	1.159	15089	99	159	2.3
03	1	30	8	1.159	17717	93	183	2.6
04	0.1	10	13	0.803	4540	34	56.9	0.8
05	0.1	20	10	0.803	6184	34	75.5	1.1
06	0.1	30	9	0.803	7543	30	93.7	1.3
07	0.01	10	14	0.803	1997	13	33.2	0.5
08	0.01	20	11	0.803	2844	15	44.8	0.6
09	0.01	30	11	0.803	3872	16	58.9	0.8
Raw				3.356	1011840	640	6976	100

TABLE III
EXTERNAL ACTIVE-SET STRATEGY WITH SNOPT, FOUR-UAV EXAMPLE

#	ϵ	N_{iter}	i_T	f^0	N_{grad}	$\lvert\mathbf{Q}\rvert$	t_C	$\%_R$
01	1	10	7	1.159	3212	77	36.9	1.6
02	1	20	7	1.159	3505	77	39.9	1.7
03	1	30	7	1.159	3646	77	41.8	1.8
04	0.1	10	9	1.158	2089	31	29.5	1.3
05	0.1	20	9	1.158	2197	31	31.3	1.4
06	0.1	30	9	1.158	2250	31	31.8	1.4
07	0.01	10	11	0.803	924	16	16.8	0.7
08	0.01	20	11	0.803	965	16	17.6	0.8
09	0.01	30	11	0.803	965	16	17.5	0.8
Raw				3.356	313600	640	2318	100

TABLE IV
EXTERNAL ACTIVE-SET STRATEGY WITH KNITRO, FOUR-UAV EXAMPLE

#	ϵ	N_{iter}	i_T	f^0	N_{grad}	$\lvert\mathbf{Q}\rvert$	t_C	$\%_R$
01	1	100	$*^1$	*	*	*	*	*
02	1	200	*	*	*	*	*	*
03	1	300	9	1.148	80614	90	892	22.6
04	0.1	100	*	*	*	*	*	*
05	0.1	200	10	0.803	21826	35	308	7.8
06	0.1	300	10	0.803	21826	35	313	7.9
07	0.01	100	17	0.803	16254	16	265	6.7
08	0.01	200	11	0.803	7733	16	142	3.6
09	0.01	300	11	0.803	7733	16	141	3.6
Raw				3.349	479360	640	3944	100

[1] '*' indicates that no reduction in computation time was achieved.

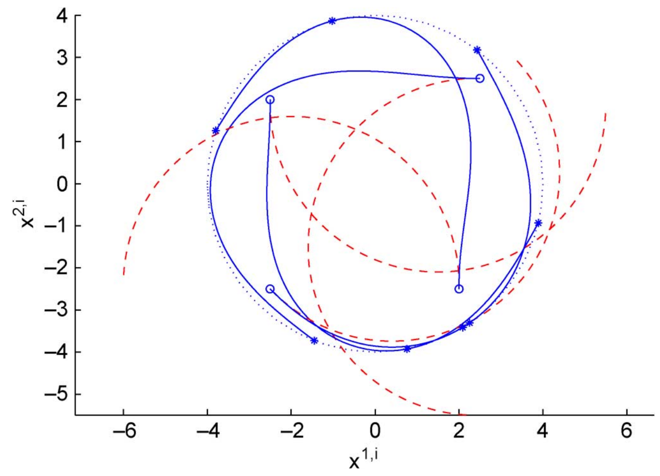

Fig. 1. Initial trajectory (dashed red) and an optimal trajectory (solid blue). Bounding circular region is represented by the dotted blue circle. Active constraints (constraints within feasibility tolerance) are marked as '*' and initial positions are marked as 'o'.

time. As can be seen from our tables, in several cases using NPSOL and SNOPT, our algorithm used less than 1/100 of the CPU time required by NPSOL or SNOPT to solve the example problem directly, i.e., using the full set of constraints.

IV. CONCLUSION

We have presented an external active-set strategy for solving discrete-time optimal control problems with state-space constraints, using nonlinear programming solvers. Our numerical results show that this strategy results in considerable savings in computer time. In our example, the savings ranged from a factor ranging from around 20 to a factor around of 135 on a problem with 640 constraints. The results depend on the nonlinear programming solver. There is reason to believe that the larger the number of inequalities in the discrete-time optimal control problem, the larger the computational savings will be. This observation is consistent with the two examples presented in this technical note. Finally, it should be obvious that one can add refinements to our algorithm, such as restarting it after a certain number of iterations, so as to avoid accumulating constraints that are not close to being active at the solution.

ACKNOWLEDGMENT

The authors wish to thank Dr. M. Saunders, Stanford University, for his advice on warm start of NPSOL and SNOPT, and M. Edvall, TOMLAB Optimization, Inc., for his advice on our codes.

REFERENCES

[1] J. T. Betts, "Survey of numerical methods for trajectory optimization," *AIAA J. Guid., Control, Dyn.*, vol. 21, no. 2, pp. 193–207, 1998.

[2] A. Schwartz and E. Polak, "Consistent approximations for optimal control problems based on runge-kutta integration," *SIAM J. Control Optim.*, vol. 34, no. 4, pp. 1235–1269, 1996.

[3] E. Polak, *Optimization: Algorithms and Consistent Approximations*, ser. Applied Mathematical Sciences. New York: Springer, 1997, vol. 124.

[4] K. Holmström, A. O. Göran, and M. M. Edvall, *User's Guide for TOMLAB*. San Diego, CA: Tomlab Optimization Inc., Dec. 2006.

[5] W. Murray, P. E. Gill, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM J. Optim.*, vol. 12, pp. 979–1006, 2002.

[6] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright, User's Guide for NPSOL 5.0: A Fortran Package for Nonlinear Programming Systems Optimization Laboratory, Department of Operations Research, Stanford University, Tech. Rep. SOL 86-2, 1998.

[7] K. Schittkowski, On the Convergence of a Sequential Quadratic Programming Method With an Augmented Lagrangian Line Search Function Systems Optimization laboratory, Stanford University, Stanford, CA, 1982, Tech. Rep..

[8] K. Holmström, A. O. Göran, and M. M. Edvall, *User's Guide for TOMLAB/KNITRO v5.1*. San Diego, CA: Tomlab Optimization Inc., Apr. 2007.

[9] C. T. Lawrence, J. L. Zhou, and A. L. Tits, User's Guide for CFSQP Version 2.5: A C Code for Solving (Large Scale) Constrained Nonlinear (Minimax) Optimization Problems, Generating Iterates Satisfying all Inequality Constraints Institute for Systems Research, University of Maryland, College Park, MD, Tech. Rep. TR-94-16r1, 1997.

[10] E. Polak, R. S. Womersley, and H. X. Yin, "An algorithm based on active sets and smoothing for discretized semi-infinite minimax problems," *J. Optim. Theory Appl.*, vol. 138, no. 2, pp. 311–328, 2008.

[11] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Boston, MA: Athena Scientific, 2003.

[12] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York: Springer, 1999.

[13] F. John, "Extremum problems with inequalities as side condtions," in *Studies and Essays: Courant Anniversary Volume*, K. O. Friedrichs, O. W. Neugebauer, and J. J. Stoker, Eds. New York: Interscience Publishers, Inc., 1948, pp. 187–204.

[14] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games," *IEEE Trans. Automat. Control*, vol. 50, no. 7, pp. 947–957, Jul. 2005.

[15] Y. Kang and J. Hedrick, "Design of nonlinear model predictive controller for a small fixed-wing unmanned aerial vehicle," in *Proc. AIAA Guid., Navig, Control Conf.*, 2006, [CD ROM].