

Decentralized Nonlinear Model Predictive Control of Multiple Flying Robots in Dynamic Environments

David H. Shim
Autonomously Controlled Advanced Platforms
(ACAP LLC)
Berkeley, CA 94709, USA
dashim@acapsys.com

H. Jin Kim Shankar Sastry
EECS Department
University of California at Berkeley,
Berkeley, CA 94720
{jin,sastry}@eecs.berkeley.edu

Abstract—In this paper, we present a nonlinear model predictive control (NMPC) for multiple autonomous helicopters in a complex environment. The NMPC provides a framework to solve optimal discrete control problems for a nonlinear system under state constraints and input saturation. Our approach combines the stabilization of vehicle dynamics and the decentralized trajectory generation, by including a potential function that reflects the state information of possibly moving obstacles or other vehicles to the cost function. We present various realistic scenarios, which show that the integrated approach outperforms a hierarchical structure composed of a separate controller and a path-planner based on the potential function method. Furthermore, the computation load of this approach is light enough to be used for the real-time control of autonomous helicopters.

I. INTRODUCTION

Autonomous helicopters, or rotorcraft-based unmanned aerial vehicles (RUAVs), have emerged as useful platforms for intelligent mobile robots due to their unique flight capabilities. Nonlinear model predictive control (NMPC) [1] is promising for systems such as RUAVs shown in Fig. 1, due to its capability to handle nonlinearity subject to operating constraints. However, the computation load of the conventional NMPC was often prohibitive for many dynamic systems with a ‘fast’ response. In [2], we applied NMPC for regulating RUAVs in the presence of input and state constraints. The minimization problem was solved with a gradient-descent method as in [3], which is computationally light and fast. The proposed NMPC was employed as a trajectory generation and tracking control layer in a hierarchical flight management system for RUAVs, and outperformed a linear controller in tracking aggressive trajectories under parametric uncertainty.

Planning a collision-free path for a robot to move from an initial to a final configuration has been the topic of extensive research as a central problem in robotics. However, many versions of this complex problem have been shown PSPACE hard [4] even in a static environment. Also, many methods work only for the discrete state space or for the special shape of obstacles [5]. In fact, deterministic performance guarantee exists only for very simplified cases (for example, see [6] for the two-dimensional aircraft with constant linear speeds and linear constraints using a centralized approach). Potential function methods [7] have dominated the obstacle avoidance



Fig. 1. Collision avoidance experiment with autonomous helicopters

research because the idea of generating a repulsive field around each obstacle is intuitive and simple to implement. It is well-known that such approaches are prone to local minima and there have been some research on generating so-called navigation functions that are free from local minima [8]. However, generating a navigation function is computationally involved and thus not suitable for many online navigation problems.

In this paper, we extend our previous work ([2]) to resolve the path planning and optimal control problems for multiple mobile robots in a complex three-dimensional environment by nonlinear model predictive control and potential function techniques. We show that our integrated approach can solve various realistic scenarios involving multiple RUAVs and obstacles in a complex three-dimensional environment. Section II presents the mathematical framework, and Section III presents the simulation results in various realistic examples involving static/moving obstacles and multiple helicopters in a three-dimensional complex environment. Section IV conclude the paper with the discussion and future directions.

II. PROBLEM FORMULATION

This section presents the formal framework for solving the path planning and optimal control problem for multiple RUAVs in a complex three-dimensional environment, using the nonlinear model predictive control and the potential function method in an integrated manner.

A. Nonlinear Model Predictive Control

In this paper, we consider a system of N (possibly heterogeneous) flying robots in a dynamic environment. The dynamics of each vehicle can be described by the following set of (Nonlinear) controlled differential equations:

$$\dot{\mathbf{x}}_i = f_i(\mathbf{x}_i) + g_i(\mathbf{x}_i)\mathbf{u}_i, \quad i = 1, \dots, N, \quad (1)$$

with the initial conditions $x_i(0)$, and where $x_i(t) \in \mathcal{X}_i$ and $u_i(t) \in \mathcal{U}_i$. Each \mathcal{X}_i is a constraint space, subset of $\mathbb{R}^{n_{x_i}}$, and each \mathcal{U}_i is the input space for the vehicle i . As we assume that the individual vehicle dynamics are decoupled, we introduce the following vector notation for the overall system;

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \quad (2)$$

where $\mathbf{x} \in \prod_{i=1}^N \mathcal{X}_i$, and $\mathbf{u} \in \prod_{i=1}^N \mathcal{U}_i$.

We are interested in solving a *decentralized discrete-time optimal control* problem for the i th subsystem, i.e. to find the optimal input sequence $\{\mathbf{u}_i^*(k)\}_{k=1}^T$ such that

$$\{\mathbf{u}_i^*(k)\}_{k=1}^T = \arg \min \sum_{k=1}^T q_i(\mathbf{x}(k), \mathbf{u}(k)) + q_{if}(\mathbf{x}(T+1)) \quad (3)$$

subject to the differential equations (1) and (2), and the terminal constraint $x(T+1) \in \mathbf{X}_f$.

Nonlinear model predictive control (NMPC) problem consists of the following steps; 1) solve for the optimal control law starting from the state $\mathbf{x}(k)$ at time k , 2) implement the optimal input $\mathbf{u}^*(k), \dots, \mathbf{u}^*(k+\tau-1)$ for $1 \leq \tau \leq T^1$, and 3) repeat these steps from the state $\mathbf{x}(k+\tau)$ at time $k+\tau$.

Often a quadratic function is used for $q_i(\mathbf{x}(k), \mathbf{u}(k))$ and $q_{if}(\mathbf{x}(T+1))$, and see [3] and [2], for a nonlinear-programming algorithm using Lagrange multipliers.

B. Model Helicopter Dynamics

Although the techniques presented in this paper are applicable to various types of mobile robots, autonomous helicopters are of our main interests. The helicopter is an under-actuated system, whose configuration space is $SE(3) \triangleq \mathbb{R}^3 \times SO(3)$ yet only four degree-of-freedom can be achieved by four inputs to the lateral cyclic pitch, longitudinal cyclic pitch, main rotor collective pitch, and tail rotor collective pitch (Eqn. (6))².

Let the superscripts S and B denote spatial and body coordinate, ϕ , θ , and ψ denote roll, pitch, and yaw, and p , q , and r are the corresponding angular rates in the body coordinate, respectively. The overall system dynamics are divided into the kinematics (Eqn. (4)) and the system-specific

dynamics (Eqn. (5)), denoted by the superscripts K and D , respectively³:

$$[\dot{x}^S, \dot{y}^S, \dot{z}^S] = R^{B \rightarrow S}[\dot{x}^B, \dot{y}^B, \dot{z}^B] \quad (4)$$

$$\dot{\mathbf{x}}_i^D(t) = f_c(\mathbf{x}_i^D(t), \mathbf{u}_i(t)) \quad (5)$$

$$\mathbf{x}_i = [\mathbf{x}_i^K, \mathbf{x}_i^D] \in \mathbb{R}^{n_x}$$

$$\mathbf{x}_i^K = [x^S, y^S, z^S, \phi, \theta, \psi]$$

$$\mathbf{x}_i^D = [u, v, w, p, q, r, a_{1s}, b_{1s}, r_{fb}]$$

$$\mathbf{u}_i = [u_{a1s}, u_{b1s}, u_{\theta_M}, u_{\theta_T}] \in \mathbb{R}^{n_u}, \quad (6)$$

where a_{1s} and b_{1s} are longitudinal and lateral flapping angles, and r_{fb} is the feedback gyro system state. A transformation matrix between the spatial and body velocities is given by $R^{B \rightarrow S} \in SO(3)$, i.e., the rotational matrix of the body axis relative to the spatial axis, represented by ZYX Euler angles $[\phi, \theta, \psi]$. The Newton-Euler equation yields the differential equation (5) for \mathbf{x}^D , which is characterized by nonlinear functions of force and moment terms. Interested readers can refer to [9] for details of the model.

C. Trajectory Generation and Tracking under Input/State Constraints

We use the following quadratic functions of state variables and inputs as the cost for tracking control of the i th helicopter at time t in Eqn. (3),

$$\begin{aligned} q_i(\mathbf{x}(k), \mathbf{u}(k)) &= q_i^{st}(\mathbf{x}_i(k), \mathbf{u}_i(k)) \quad (7) \\ &\triangleq \frac{1}{2} (\mathbf{y}_{id}(k) - C\mathbf{x}_i(k))^T Q (\mathbf{y}_{id}(k) - C\mathbf{x}_i(k)) \\ &\quad + \frac{1}{2} \mathbf{u}_i(k)^T R \mathbf{u}_i(k), \\ q_{if}(\mathbf{x}(T+1)) &= \frac{1}{2} (\mathbf{y}_{id}(T+1) - C\mathbf{x}_i(T+1))^T P_0 (\mathbf{y}_{id}(T+1) - C\mathbf{x}_i(T+1)) \end{aligned}$$

where $\mathbf{y}_{id}(\cdot)$ denotes the desired trajectory for the i th helicopter.

In order to generate the control inputs of the acceptable magnitude, input constraints are enforced by projecting each $\mathbf{u}_i(k)$ into the constraint set. In our helicopter model, this corresponds to $[u_{a1s}, u_{b1s}, u_{\theta_M}, u_{\theta_T}] \in [-1, 1]^4$. State constraints are also incorporated as an additional penalty in the cost function, i.e., the cost for the i th helicopter at time t , $q_i(\mathbf{x}(k), \mathbf{u}(k))$ of Eqn. (3), now includes

$$q_i^{sc}(\mathbf{x}_i(k)) \triangleq \sum_{l=1}^{n_x} S_{il} \max(0, |x_{i,l}(k) - x_{i,l}^{\text{sat}}|)^2, \quad (8)$$

where $x_{i,l}(k)$ denotes the l th state variable for the i th helicopter at time t , and S_{il} , and $x_{i,l}$ are constants.

¹In all examples we present, we set $\tau = 1$.

²We exclude the rotor throttle from our definition of control inputs, since we assume that the rotor rpm is maintained constant by an engine governor.

³For the notational simplicity, we will often drop the subscript i to denote the i th helicopter.

D. Decentralized collision-free trajectory generation

Our model-predictive path planning strategy adopts the idea from the potential field method [7], [4], which has been popular in path planning for mobile robots.

The cost (3) can be formulated to reflect the aspect of a potential function for path planning in the environment with moving obstacles or other agents. This allows the trajectory generation and vehicle stabilization to be combined into a single problem. In this scenario, we assume that each vehicle is aware of other vehicles' real-time location via some type of onboard sensors or inter-vehicular communication and solve decentralized optimization.

The following potential function term is added to the cost function for the helicopter j , whose position at time t is denoted by \mathbf{x}_k^j .

$$q_i^{moa}(\mathbf{x}_j(k)) = \sum_{l \neq j}^N \frac{K_{jl}}{a_j^2(x_j(k) - x_l(k))^2 + b_j^2(y_j(k) - y_l(k))^2 + (z_j(k) - z_l(k))^2}, \quad (9)$$

where $(x_l(k), y_l(k), z_l(k))$ denote the position of the helicopter l at time k , and constants a_j, b_j and K_{jl} determine the shape of repulsive potential field and thus, the adjusted trajectory. We add a repulsive potential of the same form for the moving obstacles if the environment is dynamic. Thus, the information of the other helicopters or moving obstacles, such as their position and velocity, if known, is used to *predict* the information over the next T horizon and compute the cost. In fact, this aspect contributes to the better performance of our approach over the conventional potential-function based method as discussed in Section III.

Navigation in a more complicated environment can be resolved by extending the single-point avoidance method. Finding a feasible path through obstacles may be achieved by using a sum of ellipsoidal penalties for a set of N_{pt} nearby points;

$$q_i^{terr} = \sum_{n=1}^{N_{pt}} q_i^{oa,n} \quad (10)$$

where $J_i^{oa,n}$ is the penalty on the distance from a n -th nearest obstacle to the i -th helicopter, similar to the one given in (9).

We apply this method to the urban navigation problem in Section III-B. The distance to the selected points from the UAV is used to compute the potential function at every future position along the finite horizon in the optimization. A large value of N_{pt} will reduce sudden jumps in the cost function. However, penalizing the distance to the single nearest point turns out to be computationally light yet effective enough in the examples we tried. In actual experiments, finding the closest point can be done by an onboard laser scanner or an ultrasonic sensor array.

E. Three-dimensional pursuit-evasion game

In this case, we consider two UAVs in a close-range air operation. One UAV is in pursuit of the other UAV. The

pursuing UAV's goal is to align its heading to the target UAV and reduce the distance without crashing into the target. The evading UAV tries to escape from the point where it is aligned to the heading of the pursuing UAV. This can be formulated as two separate objectives, which are not necessarily in conflict: 1) align its heading to the target vector $\mathbf{x}_{E \leftarrow P}$ and 2) avoid being aligned in the target's heading \mathbf{x}_E^H , as illustrated in Fig. 2. For the pursuer, the deviation of the relative heading angle between $\mathbf{x}_P^H \triangleq R_P^{B \rightarrow S}[1, 0, 0]^T$ and the relative position vector $\mathbf{x}_{E \leftarrow P}$ is penalized to obtain the best aim. For the evader, the cost as a function of α_D is formulated to be zero when $\alpha_D = \pm 90^\circ$ and to be maximum at 0° and 180° . In addition to these penalties on headings, the relative distance between the pursuer and evader is also included as a part of their cost function. For the pursuer, the distance $\mathbf{x}_{E \leftarrow P}$ is penalized for more effective pursuit. The relative distance is inversely penalized in the cost function of the evader for exactly the opposite reason.

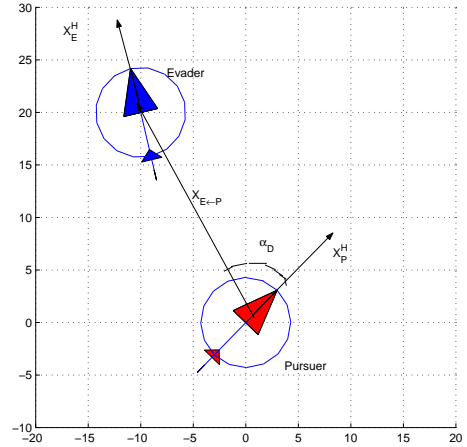


Fig. 2. pursuit-evasion game, where a pursuer wants to minimize the relative angle and an evader wants to maximize it.

To summarize, the following cost functions are proposed for the pursuer and the evader, respectively;

$$q_p(\mathbf{x}_P(k), \mathbf{x}_E(k)) \quad (11)$$

$$= K_p \mathbf{x}_{E \leftarrow P}^T (K_{dist} I - \text{sgn}(\mathbf{x}_{E \leftarrow P} \cdot \mathbf{x}_P^H) \mathbf{x}_P^H \cdot (\mathbf{x}_P^H)^T) \mathbf{x}_{E \leftarrow P}$$

$$q_e(\mathbf{x}_P(k), \mathbf{x}_E(k)) \quad (12)$$

$$= K_e \mathbf{x}_{E \leftarrow P}^T (\mathbf{x}_P^H \cdot (\mathbf{x}_P^H)^T) \mathbf{x}_{E \leftarrow P}.$$

The overall cost function term for each player is now given by

$$q_i(\mathbf{x}(k), \mathbf{u}(k)) \quad (13)$$

$$= q_i^{st}(\mathbf{x}_i(k), \mathbf{u}_i(k)) + q_i^{sc}(\mathbf{x}_i(k))$$

$$+ q_i^{bdry}(\mathbf{x}_i(k)) + q_i^{moa}(\mathbf{x}(k))$$

$$+ q^p(\mathbf{x}(k)) + q^e(\mathbf{x}(k)).$$

⁴Note that the *heading* here is not same as the *yaw* angle, as we are treating the three-dimensional dynamics.

In addition to the two usual cost function terms q^{st} and q^{sc} , q^{bdry} , q^{moa} , q^p and q^e are included in (13). q^{bdry} is the cost function in the form given in Eqn. (10), which enforces the helicopters to stay in the predefined region. If this function is not included, the evading agent may fly indefinitely away in any direction to flee from its pursuer. q^{moa} is the cost function for preventing collision between two agents. Without this term, the penalty on $\|\mathbf{x}_{E \leftarrow P}\|$ in q^p may lead the pursuer to crash into the other agent. The cost functions for pursuit and evasion may be introduced together as a sum of pursuing part and evading part, as in Eqn. (13). Two extreme cases are pure pursuit and pure evasion. A pursuer persistently follows the target even if it is exposed to a higher risk from the other agent. A pure evader only avoids being aligned along the pursuing agent's line of sight. We will show the related examples in Section III-C and III-D.

III. SIMULATION RESULTS

In this section, we evaluate the effectiveness of the nonlinear model predictive trajectory planning and tracking controller proposed in Section II in various scenarios.

A. Collision-avoidance Planning and Control under Input/State Constraints

In this example, five helicopters are originally given straight-line desired trajectories that will lead to a mid-air collision as shown in Fig. 3(a). The potential function term shown in Eqn. (9) is added into the cost function of each helicopter, to resolve the confliction. In order to generate a plausible control input while trying to avoid the collision, the input saturation conditions were enforced. Also included in the cost function is the state constraints in the form of Eqn. (8), with $[\phi_{sat}, \theta_{sat}, u_{sat}, v_{sat}, w_{sat}, p_{sat}, q_{sat}, r_{sat}] = [\pi/6, \pi/6, 16.7, 16.7, 16.7 \text{ ft/s}, \pi/2, \pi/2, \pi/3 \text{ rad/s}]$ in order to contain the overall vehicle response at an acceptable range. Fig. 3(b) shows the resulting trajectories that each helicopter actually achieved.

Fig. 4 shows that our integrated approach outperforms the purely potential-function method, when the potential function is employed as a path planning layer separate from a vehicle stabilization layer in a simple collision-avoidance problem. The helicopter heading to the left is controlled by the NMPC, whereas the helicopter flying to the right, assumed to have already stabilized linear dynamics, uses the potential function method as a trajectory generation layer. When the potential function is considered in generating the trajectory only for each time step, it caused abrupt change (Point A), slow convergence back to the desired trajectory (Point B) and overshoot (Point C) in the vehicle response. On the other hand, the NMPC-based method resulted in smooth trajectory without overshoot.

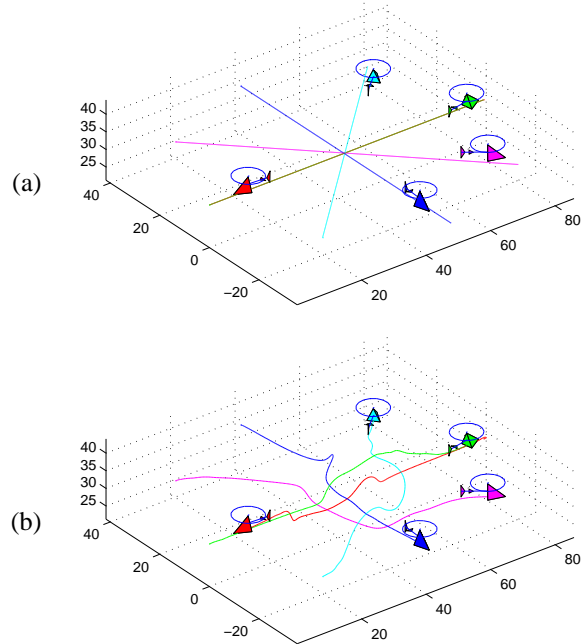


Fig. 3. Distributed collision avoidance: (a) initial configuration and the destination of each helicopter, and the corresponding desired trajectories (b) their trajectories executed

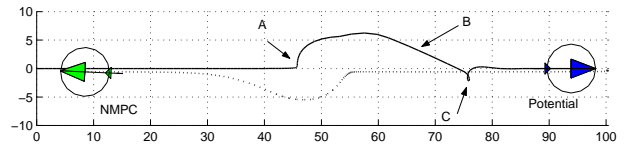


Fig. 4. Nonlinear model predictive control (dotted) vs. potential-function approach (solid)

B. Flying in a complex environment

In this example, we apply the NMPC framework for a navigation problem in which a UAV is requested to fly through a space full of building-like obstacles. This type of situation often arises in a urban area, filled with buildings of various sizes.

For this simulation, a realistic three-dimensional map filled with buildings of random width and height is generated as shown in Fig. 5(a). Then a UAV is requested to fly from the rooftop of the building at lower-left on the map to the building at the diagonal side. The NMPC-based flight controller is only given *a priori* with a simple straight flight path that directly connects the starting and destination points, and this trajectory intersects with a number of buildings along the path. The vehicle resolves the collision by maintaining the safe distance from the nearest point from nearby buildings as it travels. Fig. 5(a) and (b) shows the output of a simulation. The short lines from the UAV's trajectory to the nearby buildings indicate the vector of the minimum distance

obtained from the scanning process. Along the path, the UAV encounters a number of buildings which are too close to itself. The UAV successfully reaches the destination by detouring the building along the given path.

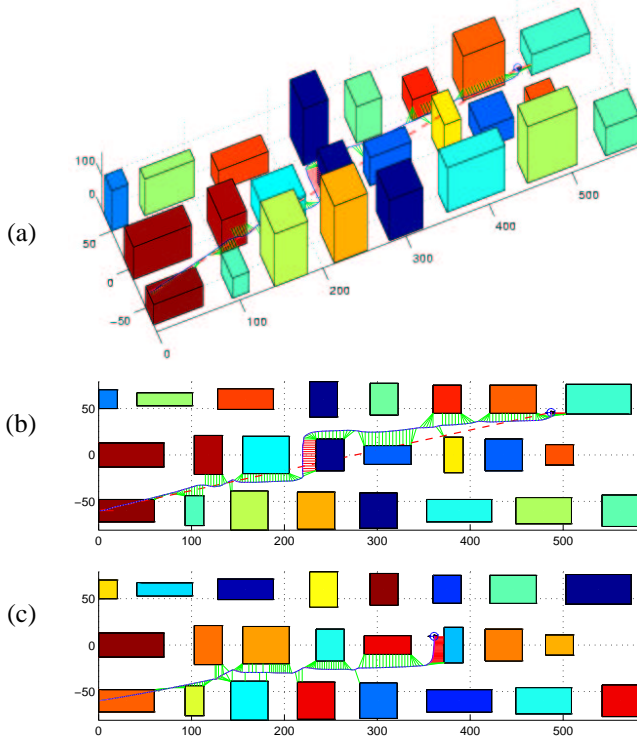


Fig. 5. Flying in a complex 3-D environment using (a),(b): NMPC, and (c): potential-function approach. (b) and (c) are top-down views.

Although considering N_{pt} nearest points, not just one, should reduce of will reduce sudden jumps in the cost function, penalizing the distance to the single nearest point turns out to be computationally light yet effective enough in this example.

Fig. 5(c) shows the result when a potential function method was applied to the same example as in Fig. 5(b). Here, we assumed that a separate lower-level stabilization layer exists and used the potential function as a trajectory generator. This result confirms the well-known fact that the potential-function method is prone to local minima. By comparing Fig. 5(b) and Fig. 5(c), we can see that our approach is less susceptible to the local minima problem due to its predictive nature.

C. Three-dimensional pursuit-evasion game

This example considers two UAVs in a close-range operation. Suppose that one UAV is in pursuit of the other UAV. The pursuing UAV's goal is to align its heading to the target UAV and reduce the distance without crashing into the target. The evading UAV's goal is to stay out of the line of sight of the pursuer as explained in Section II-E.

In Fig. 6, trajectories and snapshots from a six-second interval during a pursuit-evasion game between two full-time opposite-trait agents are shown. The pursuer attempts to shorten the distance and align its heading to the direction where the other agent is located simultaneously. The evader, on the other hand, moves away from the pursuer while refraining to show its tail to the pursuing agents.

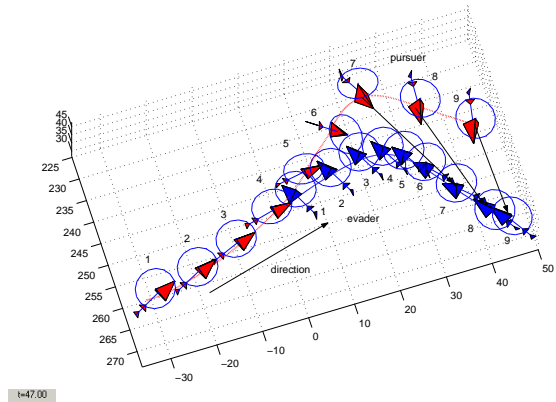


Fig. 6. Pursuit-evasion in a three-dimensional environment: lines between the pursuer and the evader represent the corresponding positions at each time instant.

D. Combining all together: pursuit-evasion game in a confined space with urban obstacles

This example, shown in Fig. 7, combined obstacle-avoidance and pursuit-evasion feature as well as the constraints on the magnitude of state variables and inputs. The UAVs in this environment are performing both pursuit and evasion as the cost function in Eqn. (13) dictates, while avoiding collision into the buildings in the confined area. As is in the scenario in Section III-B, the UAVs are successfully confined in the pre-specified region by J^{bdry} .

E. Implementation

Horizon length N , and the number of iterations during optimization, as well as the weighting matrices P_0 , Q , and R , are important design parameters related to the simulation speed and closed-loop stability. Step-size Δ_k should be carefully chosen to consistently reduce the cost during the iteration. We selected $N = 25 \sim 30$, and $\Delta_k = 0.0001 \sim 0.001$ for the simulations presented in this paper.

Our NMPC algorithm is written in CMEX format for enhanced computation speed and displayed in MATLAB. Table I summarizes the computation time for the examples shown in Sections III-A – III-D. All the examples were run in a decentralized manner for each helicopter but on a *single* Windows-XP notebook computer with a Pentium-III 1 GHz processor to generate control inputs at 50 Hz. Therefore, all the examples presented here run significantly faster than real-time, and the proposed NMPC algorithm is promising for

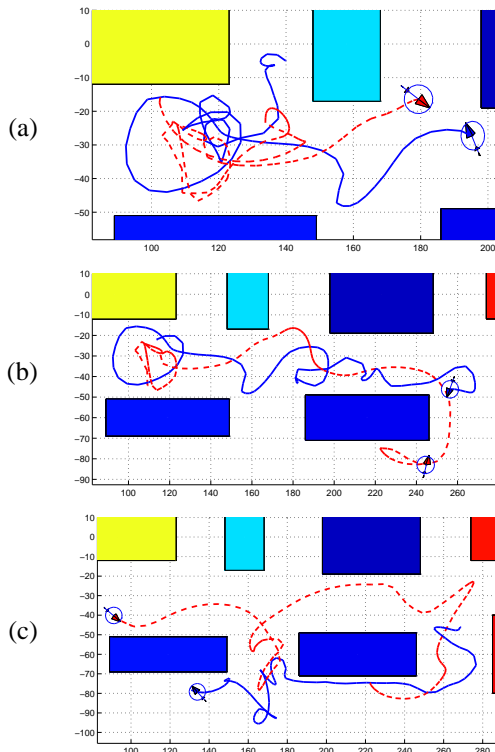


Fig. 7. Top-down view during a symmetric pursuit-evasion game in a complex three-dimensional environment: both players are pursuing and evading at the same time

Example	Flight time (sec)	Computation time (sec)	Number of helicopters
A	24	41	5
B	176	110	1
C	206	168	2
D	200	173	2

TABLE I

COMPUTATION TIME FOR THE EXAMPLES SHOWN IN SECTION III, RUN ON A *single* NOTEBOOK WITH A PENTIUM-III 1GHZ PROCESSOR

the UAV flight control systems where the host vehicle is demanded to operate in a complicated environment.

Although deliberation of the optimality over the time horizon T renders our NMPC-based approach less prone to the local minima than purely potential-function-based methods, we would need T to be large enough to *foresee* over the local minima. Since the computation time directly depends on T , it would be more efficient to combine some high-level switching logic to determine a suitable value for T depending on the type of operation, rather than blindly increasing T .

IV. CONCLUSION AND FUTURE WORK

In this paper, we formulated a nonlinear model predictive control (NMPC) framework to control multiple autonomous

vehicles in a complex three-dimensional space. Our approach combines the stabilization and trajectory generation problems, by including the potential function in the cost function. We implemented the proposed NMPC algorithm as an online decentralized optimization controller and evaluated in various realistic scenarios. In the examples shown in this paper, the state constraints were included into the cost, and the input saturation conditions were enforced to show the viability of our approach to control multiple UAVs. Although there exists no formal guarantee of the uniqueness of the minima, we observed that our approach is less prone to the local minima than the conventional potential-function methods due to the longer-term preview. The effect of tuning these parameters and adjustment of potential function to also reflect the type, speed or heading of objects require further investigation. The computational load of our NMPC formulation using the gradient-descent or conjugate gradient method is low enough to be used for controlling UAVs real-time. Some hardware experiments are currently being performed [10] and will be reported in near future.

V. REFERENCES

- [1] F. Allgöwer and A. Zheng, editors. *Nonlinear Model Predictive Control*, volume 26 of *Progress in Systems and Control Theory*. Birkhäuser Verlag, Basel-Boston-Berlin, 2000.
- [2] H.J. Kim, D.H. Shim, and S. Sastry. Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles. In *American Control Conference*, Anchorage, AK, May 2002.
- [3] G.J. Sutton and R.R. Bitmead. Computational implementation of NMPC to nonlinear submarine. In F. Allgöwer and A. Zheng, editors, *Nonlinear Model Predictive Control*, volume 26, pages 461–471. Birkhäuser, 2000.
- [4] J.C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.
- [5] A. Zelinsky. A mobile robot navigation exploration algorithm. *IEEE Transactions of Robotics and Automation*, 8(6):707–717, 1992.
- [6] A. Richards and J.P. How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *American Control Conference*, Anchorage, AK, May 2002.
- [7] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1):90–98, 1986.
- [8] E. Rimon and D.E. Koditschek. Exact robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, Oct. 1992.
- [9] D.H. Shim. *Hierarchical Control System Synthesis for Rotorcraft-based Unmanned Aerial Vehicles*. PhD thesis, University of California at Berkeley, 2000.
- [10] <http://www.eecs.berkeley.edu/~jin/research.html>.