2007 Mediterranean Conference on Control and
Automation, July 27 - 29, 2007, Athens - Greece

MPC Based Motion Control of Car-like Vehicle Swarms

Wei Xi
Western Digital Corporation
Lake Forest, CA 92630, USA
Email: wei.xi@wdc.com

John S. Baras
Institute for Systems Research and
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742, USA
Email: baras@isr.umd.edu

***Abstract*— In this paper we study low level motion control of car-like vehicles for applications of UAV swarms. First, a Suboptimal Continuous-Curvature trajectory generation approach is studied to generate suboptimal reference trajectories ('tracks') to connect desired way-point sequences. Two Model Predictive Control (MPC) based 'track following' control approaches are then proposed to deal with the multiple constraints present in practice; for example, actuator saturation, local collision avoidance. Simulations further confirm the motion control design.**

I. Introduction

There has been an increasing interest in controlling swarm-based unmanned autonomous vehicles (UAVs)as the technology provides great potential to perform dangerous or explorative tasks in various hazardous, unknown or remote environments. Originally driven by the need for saving labor costs and protecting personnel loss from dangerous environments, the applications of UAVs have been extended to a broad range of problems both in military and commercial/industrial domains: automated highway systems, mobile sensor networks, ocean resources exploration, and robotic border patrol [1], [2].

Traditional centralized control approaches are not preferred because communication and computation costs increase exponentially with the size of UAV swarms. Decentralized/distributed control approaches are appealing considering the large scale of vehicle networks[3], [4]. In our earlier work [5], a Gibbs sampler based simulated annealing algorithm has been proved to be useful for autonomous vehicles achieving self-organization.

Our previous work concentrated only on the high level path planning algorithm, where a point mass model was used to represent each UAV. In this paper, a two level hierarchical scheme is adopted in the collaborative UAV swarms control system design (Figure 1). The high level path planning module generates a sequence of way-points for each UAV. By following the way-points, swarm UAVs coordinate their actions to achieve desired collective behaviors. The low level motion control computes the control commands for real UAVs to follow the way-points under the vehicle's dynamic and kinematic constraints.

This research was supported by the Army Research Office under the ODDR&E MURI01 Program Grant No. DAAD19-01-1-0465 to the Center for Networked Communicating Control Systems (through Boston University), and under ARO Grant No. DAAD190210319.

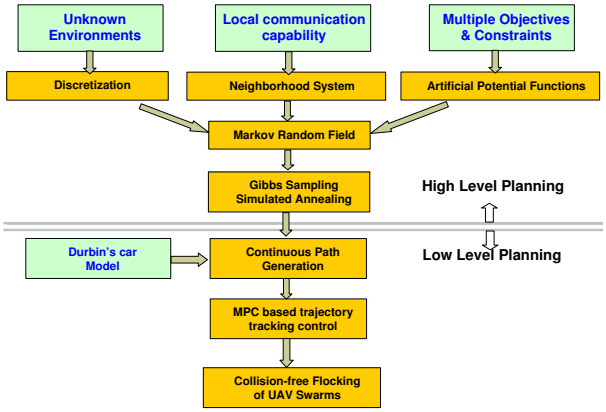

Fig. 1. Block diagram of two level hierarchical swarm UAV collaborative control: distributed Gibbs sampler and NMPC based.

In this paper, the motion control of car-like UAVs is investigated. Control of wheeled mobile robots with nonholonomic constraints has been considerably studied by the robotics and control communities [6], [7], [8]. Some practical concerns, like input/actuator saturation have also been addressed [9]. However, in applications of UAV swarms, multiple objectives, including but not limited to local traffic and obstacle avoidance, present new challenges for the motion control design. Moreover, low computational cost algorithms are more appealing to save power and cost in such applications. Inspired by the work in [10], [11], a Nonlinear Model Predictive Control (NMPC) approach is proposed to systematically address the multiple objectives oriented tracking control problem. Two variations, gradient descent (GD) and Dynamic Programming (DP), are investigated in section V. The GD based approach provides better control accuracy, but suffers from convergence difficulties and large computation time variation. The DP based approach trades control performance with computation time. The "curse of dimensionality" of DP limits the control accuracy. These considerations frame the trade offs between the two approaches. Simulation results are performed to compare and verify the proposed approaches in section VI.

II. General Problem Formulation

In [5], a path planning problem for the self-organization, in a battle field scenario, of a swarm of

UAVs is studied. A 2D mission space is first discretized into a lattice of cells. N point mass UAVs are moving on the grid lattice. Label each cell with its center coordinates (x, y). The Gibbs sampler based algorithm generates a sequence of way-points $\{p_i(k), i = 1, ..., N\}$ for each UAV i, on the fly. The sequence of way-points are then used to generate smooth continuous curves $\{q_i^r(t)\}$ which pass from the way-point $p_i(k)$ at time $t(k)$. The continuous curve $\{q_i^r(t)\}$ for UAV i is usually called a *reference trajectory* (or *track*). One can imagine $\{q_i^r(t)\}$ as the *image* of the discrete *path* $\{p_i(k)\}$ in the continuous mission space.

In practice, however, dynamic and kinematic constraints, e.g., nonholonomic constraints and input/output constraints, prohibit autonomous vehicles from following arbitrary *reference trajectories*. So the *reference trajectories* have to be generated by solving the vehicles' dynamic and kinematic differential equations, which is considered the first problem in the low level motion control design.

$$\begin{array}{ll} Solve & q_i^r(t) \quad \forall i, i = 1, \ldots, N \\ s.t. & q_i^r(t(k)) = p_i(k) \\ and & \dot{q}_i^r(t) = f(q_i^r(t), \dot{q}_i^r(t)) \end{array} \tag{1}$$

where $q_i^r(t)$ is the *reference trajectory* of the generalized coordinate q for vehicle i, and the last equation is the vehicle's kinematic differential equation. It has to be noted that the solution of the *reference trajectory* $q_i^r(t)$ might not be unique. In section IV, we focus on a specific time optimal trajectory generation problem for car-like vehicles.

With $q_i^r(t)$ available, the remaining problem is to design a track following control law such that a real vehicle can follow the desired *reference trajectory* (or *track*). As aforementioned, practical concerns, for example, actuator saturation, and local traffic/collision avoidance, have to be addressed in the low level track following control design. To be specific, the problem can be formulated as follows:

$$\begin{array}{ll} Design & u_i(t), \quad \forall i = 1, \ldots, N \\ s.t. & |q_i(t) - q_i^r(t)| \le d_i \\ and & g_j(x_i(t), \dot{q}_i(t), u_i(t)) \le 0, \quad \forall j = 1, \ldots, M, \end{array} \tag{2}$$

where $u_i(t)$ is the control command, $g_j(\cdot) \le 0$ are the set of input/output/state constraints.

III. Kinematic Model of Car-like UAV

In this paper, a rear-wheel driving car-like UAV is adopted due to its relative lower cost and convenience for applications. However, the presence of nonholonomic constraints in its kinematic model, which usually refers to the *rolling without slipping* constraint between the wheels and the ground, impose many difficulties in control design. In particular, Brockett [12] showed that a linearized nonholonomic model has deficiency in controllability and there is no time-invariant linear control to guarantee the tracking error convergence to zero.

The configuration of the rear-wheel driving vehicle is shown in Figure 2. In this model, the generalized coordinate $q = (x, y, \theta, \phi)$, where (x, y) are the cartesian

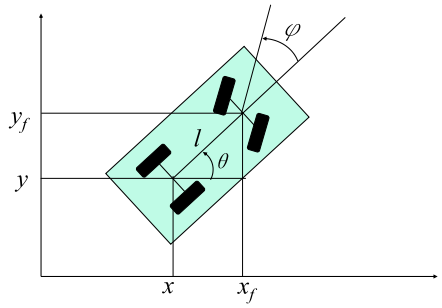

Fig. 2. The rear-wheel driving car model.

coordinates of the center point of the rear axle, θ is the heading angle of the car body with respect to the x axis, and ϕ is the steering angle. In Figure 2, l is the distance between the front axle and the rear axle. Let (x_f, y_f) be the coordinates of the front axle center point. Assuming the two front and back wheels are parallel, the nonholonomic constraints can be expressed as follows:

$$\begin{aligned} \dot{x}_f \sin(\theta + \phi) - \dot{y}_f \cos(\theta + \phi) &= 0 \\ \dot{x} \sin\theta - \dot{y} \cos\theta &= 0. \end{aligned} \tag{3}$$

Considering the rigid body constraints, the center of the front axle (x_f, y_f) satisfies

$$\begin{bmatrix} x_f \\ y_f \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} l + \begin{bmatrix} x \\ y \end{bmatrix}. \tag{4}$$

The first nonholonomic constraint (3) can be rewritten with only the general configuration q involved:

$$\dot{x} \sin(\theta + \phi) - \dot{y} \cos(\theta + \phi) - \dot{\theta} l \cos\theta = 0. \tag{5}$$

Assuming that the control inputs v, ω are linear velocity and steering velocity respectively, the kinematic model of a rear-driving vehicle can be expressed as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ \tan\phi/l & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \tag{6}$$

One can easily verify that (6) incorporates the nonholonomic constraints (3) and (5). Note that when $\phi = \pm\frac{\pi}{2}$, the model becomes singular. This corresponds to the situation where the front wheel heading is orthogonal to the car longitudinal axis. In practice, the range of the steering angle ϕ is restricted to prevent this singular case.

IV. Trajectory Generation for Car-like UAV

In this section, the generation of the continuous *reference trajectory* is investigated for car-like vehicles given predetermined way-point sequences in free space. One of the most interesting problems in the literature is the shortest path problem, which is usually associated with the time-optimal trajectory. However, nonholonomic constrains of the car-like vehicle present difficulties in solving such kind of problems.

In 1957, Dubins studied the problem for the unicycle model with constant linear velocity 1 [13]. He showed that the optimal trajectories are concatenations of at most

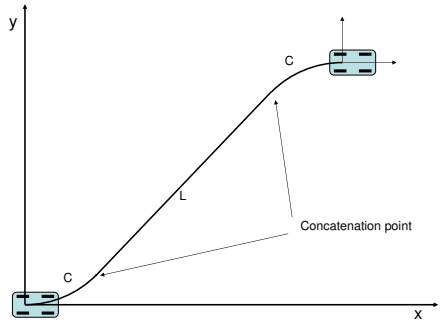

Fig. 3. An example of the optimal trajectory for Dubins' car

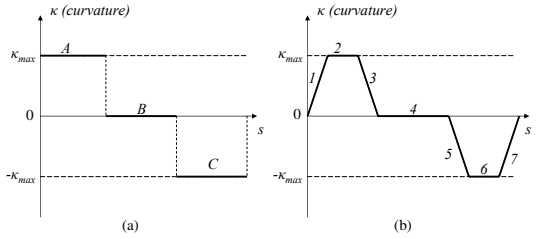

Fig. 4. Curvature profiles: (a) Dubins' trajectory and (b) SCC trajectory

3 pieces of basic elements, which include a line segment (L) and an arc (C) with radius 1. A typical example of the optimal trajectory for Dubins' car is shown in Figure 3, where a C-L-C type trajectory is used to connect the initial and terminal configurations.

However, in Dubins' optimal trajectory, there may exist discontinuous curvatures at the connection point of two successive pieces, for example, line-arc or arc-arc (with opposite direction of rotation). To follow (exactly) such a trajectory a car-like vehicle would be constrained to stop at the end of each connection point. To deal with the problem, Sussmann [13] studied a generalization of the problem by controlling the angular acceleration instead of the angular velocity, which is an equivalent model of the car-like vehicle. He showed that the optimal trajectory consists of line segments, arcs and clothoids. Moreover, Sussmann showed that in some extreme cases, there may exist a time-optimal trajectory that involves infinite chattering [13], which is not allowed in practice.

A practical way for addressing the problem is to generate sub-optimal trajectories by restricting the maximum number of connection points. The analytical study of the Sub-optimal Continuous-Curvature (SCC) trajectory planning can be found in [14]. In the SCC trajectory, there exist at most 9 pieces of basic elements. For each Dubins' optimal trajectory there is a corresponding sub-optimal trajectory. For example, the C-L-C in Dubins' model may become Cl-C-Cl-L-Cl-C-Cl in the sub-optimal trajectory. The curvature profile for the example of Figure 3 is shown in Figure 4. By replacing the arcs A and C in the left plot with the curves 1-2-3 and 5-6-7 in the right plot, the curvature profile is continuous. The key part of this approach is to replace any arc segment in Dubins' optimal trajectory with a *continuous-curve-turn*. More precisely, the arc is replaced by a Cl-C-Cl combination, where the

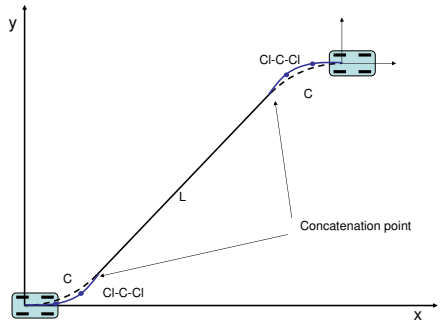

Fig. 5. An example of the sub-optimal trajectory

start and end points have curvature equal to 0 [14].

In general, the local sub-optimal trajectory planner works as follows. First, generate the Dubins' optimal trajectory using the synthesis approach in [15]. For each arc segment, replace it by a curve consisting of Cl-C-Cl. A typical sub-optimal trajectory for the example of Figure 3 is shown in Figure 5. In [16], Scheuer further compared the Dubins's optimal trajectory and the SCC trajectory. In the extensive simulation results of [16], it was shown that the total length of the SCC trajectory is only about 1.1 times longer than the Dubins' optimal trajectory. In the rest of this paper, it is assumed that the reference trajectory $q_r(t)$ is generated by the sub-optimal method. The corresponding reference control inputs are linear velocity $v_r(t)$ and steering velocity ω_r.

V. Model Predictive Control (MPC) Based Trajectory Tracking Control

A. MPC Based Motion Control

As aforementioned, the low level motion control module has to address multiple constraints besides track following control; for example, local collision avoidance, local obstacle avoidance, input/state saturation, etc.. In the literature, feedback control approaches for car-like vehicles have been proposed by many researchers to achieve zero tracking error [6], [17]. However, none of them addressed the problem stated in 2. Inspired by the work in [10], [11], MPC based motion control approaches are studied in this section.

The general framework of the MPC approach is depicted in Figure 6. The main idea of the MPC approach is to choose the control action by repeatedly solving online an optimal control problem.

More precisely, in discrete time, the model predictive control approach can be formulated as follows:

$$\begin{aligned} x(t+1) &= f(x(t), u(t), w(t)) \\ y(t) &= g(x(t), u(t)), \end{aligned} \tag{7}$$

where $u(t)$ is the control input, and $w(t)$ is the noise or disturbance. At time$t$, given a control input sequences $\{u_k\}_t^{t+N}$ and the initial state $x(t)$ at time instant t, for a finite time horizon $H = \{t, ..., t+N\}$, the output $\{y_k\}_t^{t+N}$ can be calculated by the predictive model (7). Let us define a cost/objective function as $J(\{x_k\}_t^{t+N}, \{y_k\}_t^{t+N}, \{u_k\}_t^{t+N})$ involving the future state trajectory, output trajectory and control effort. The

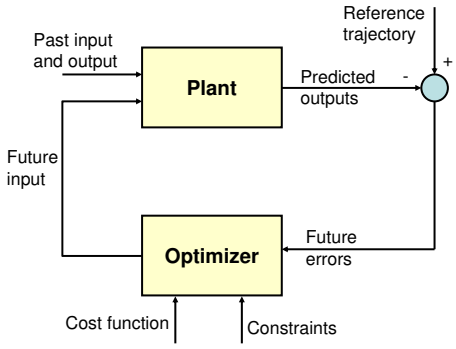

Fig. 6. Basic structure of Model Predictive Control (MPC)

optimal control input u^* is generated by minimizing the cost function, that is,

$$\{u_k^*\}_t^{t+N} = \arg \min_{\{u_k\}_t^{t+N}} J(\{x_k\}_t^{t+N}, \{y_k\}_t^{t+N}, \{u_k\}_t^{t+N}). \tag{8}$$

From time instant t to $t+\tau$, the optimal control inputs $\{u_k^*\}_t^{t+\tau}$ are then applied to the process or plant. At time instant $t+\tau$, a new time horizon $T = \{t+\tau, ..., t+N+\tau\}$ is generated. $\{u_k^*\}_{t+\tau}^{t+N+\tau}$ is calculated by solving a similar finite horizon optimal control problem as (8), given $x(t+\tau)$ as initial state and the predictive model (7). $\{u_k^*\}_{t+\tau}^{t+2\tau}$ are then used for control during the time interval $\{t+\tau, ..., t+2\tau\}$. One can imagine the finite horizon as a window with size N moving along the time axis (rolling horizon). Recursively solving a finite horizon optimal control problem with (time horizon) length N and applying the computed optimal control during the subsequent τ steps is essentially the core of the MPC strategy.

B. Encode Objectives and Constraints

In the MPC approach, multiple objectives and constraints can be encoded in the objective function J. To be specific, in our problem, the following objective function is used:

$$J = \psi_{t_f} + \int_{t_0}^{t_f} (\mu^{tk} J^{tk} + \mu^{sc} J^{sc} + \mu^u J^u + \mu^o J^o + \mu^c J^c) dt, \tag{9}$$

where J^{tk}, J^{sc}, J^u, J^o, and J^c are the objective/potential functions accounting for tracking performance, state saturation, control effort and saturation, obstacle avoidance, and collision avoidance respectively [18]. μ^{tk},μ^{sc},μ^u,μ^o, and μ^c are weighting coefficients for each objective. The design of the potential functions and weighting coefficients are challenging in order to get robust performance. In our algorithm, the potential functions are designed as follows:

- Tracking performance J^{tk}
  Assume that the desired trajectory is $q_d(t) = \{[x_d(t), y_d(t), \theta_d(t), \phi_d(t)]\}$. The quadratic form of tracking error can be a good candidate, i.e.,
  $$J^{tk} = (q - q_d)^T Q (q - q_d),$$
  where Q is a positive diagonal matrix.
- Terminal cost ψ_{t_f}
  Similarly as with the tracking performance, ψ_{t_f} is selected as:
  $$J^{tk} = (q_{t_f} - q_d(t_f))^T Q_0 (q_{t_f} - q_d(t_f)),$$
- Input/state saturation J^{tk}
  As noted, when the steering angle $\phi = \pm \pi/2$, the kinematic model of a rear-wheel vehicle will degenerate. This introduces the need to enforce that the steering angle must live in a 'safe' range $[-phi^{sat}, phi^{sat}]$, where phi^{sat} is a positive scalar. The saturation cost could be
  $$J^{sc} = max(0, |\phi| - \phi^{sat})^2,$$
- Control effort J^u
  We use a quadratic form to measure control effort:
  $$J^u = u^T R u,$$
- Obstacle avoidance J^o
  A repulsive potential function is used for avoiding obstacles. Assume that the closest point on the obstacle surface to the vehicle is (x^o, y^o). We used a J^o having the following form
  $$J^o = \frac{1}{(x - x^o)^2 + (y - y^o)^2},$$
  where (x, y) is the location of the vehicle.
- Collision avoidance J^c
  $$J^c = \sum_j \frac{1}{\max(\sqrt{(x - x^j)^2 + (y - y^j)^2} - R_{safe}, \epsilon)},$$
  where R_{safe} is the safety range and ϵ is a small scalar used to prevent J^o from becoming infinite or negative.

C. Gradient Descent Based MPC approach

Inspired by [10], a gradient descent approach is proposed to recursively solve the finite horizon optimal control problem. The goal of the problem is to find the optimal control input $u^*(t) \in \mathcal{U}$, $t \in [t_0, t_f]$, to

$$\begin{aligned} \text{minimize } J &= \psi(q(t_f)) + \int_{t_0}^{t_f} L(q(t), u(t), t) dt \\ \text{subject to } \dot{q} &= f(q(t), u(t), t) \\ q(t_0) &= q_0 \end{aligned} \tag{10}$$

where the first term $\psi(q(t_f))$ is the terminal cost, and the second term is the running cost. By introducing the costate vector $\lambda(t)$, the Maximum Principle indicates that the optimal control should satisfy the following conditions:

$$\begin{aligned} L_u + \lambda^T f_u &= 0 \\ L_q + \lambda^T f_q + \dot{\lambda}^T &= 0 \\ \psi_q(q_{t_f}) - \lambda^T(t_f) &= 0 \end{aligned} \tag{11}$$

where L_u, L_q denote the partial derivative of L with respect to u and q, respectively. Similarly for f_u and f_q. One can find that the costate propagates backwards in time with initial condition $\lambda^T(t_f) = \psi_q(q_{t_f})$, whereas the state propagates forwards in time. This fact presents

difficult challenges to solving for the optimal control $u^*(t)$ analytically. Instead, numerical methods have been proposed that employ the gradient descent method. Such a method is outlined below:

1) For a given q_0, pick a control history $u^0(t)$. Let $i = 0$.
2) Propagate $\dot{q} = f(q, u, t)$ forward in time to create a state trajectory.
3) Evaluate $\lambda^T(t_f) = \psi_q(q_{t_f})$, and solve backwards for λ^T using $\dot{\lambda}^T = L_q + \lambda^T f_q$
4) Update the control input $u^{i+1} = u^i + \delta_u$ with $\delta_u = -K(L_u + \lambda^T f_u)$, where K is a positive scalar.
5) Calculate $\delta J = J(u^{i+1}) - J(u^i)$. If $\delta J > 0$, reduce K and go back to step 4.
6) Let $i = i+1$, and go back to step 2 until the solution converges.

This numerical method is widely used for solving optimal control problems in complex systems. Its major difficulty is the computational cost. In order to speed up solution convergence, one should carefully select an appropriate control trajectory to start the iterations. Once the optimal solution is available, one can then plug the nonlinear optimizer in the general MPC framework of the previous section to get the NMPC approach.

It has to be noted that, this method requires L to be differentiable. However, non-differentiable objective functions such as the obstacle/collision avoidance and the saturation functions present difficulties in computing the partial derivatives L_u and L_q. In practice, numerical approximations can be used for computing these derivatives. One may also use curve fitting technics, e.g., high order polynomial functions, to approximate the non-differentiable objective functions with differentiable ones.

It is straightforward to extend the approach to discrete time nonlinear systems in order to implement it on a low-cost digital controller; one may refer to [10] for details.

D. Dynamic Programming Based NMPC Approach

In section V-C, a GD approach was proposed to solve the nonlinear finite horizon optimal control problem in the NMPC approach. It has to be noted that the convergence of the GD approach is not guaranteed. Without carefully selecting the weighting coefficients in the objective function, as well as the initial control sequence and step size of control updates, this approach may lead to unstable performance. The other difficulty of the GD approach is that the computation time in different sampling periods may vary a lot, which may cause instability due to the maximum delay caused by the computations.

To address these problems, a dynamic programming (DP) approach is developed here. It is well known that the DP approach suffers from "curse of dimensionality" in general. However, since it is usually assumed that autonomous vehicles have only limited actuation capabilities, by reducing the size of the set of admissible control inputs, the DP approach can be used to solve the finite horizon optimal problem (12) in a reasonable time.

For the discrete time case, the finite horizon optimal problem (12) can be rewritten as follows

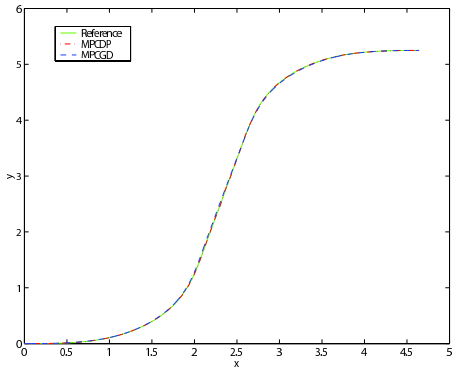

Fig. 7. Free space trajectory tracking with the MPC approach

$$\begin{aligned} \text{minimize } J &= \psi(q(N)) + \sum_{k=0}^{N-1} L(q(k), u(k)) \\ \text{subject to } q_{k+1} &= g(q(k), u(k)) \\ q(0) &= q_0. \end{aligned} \tag{12}$$

We assume that the control input $u(k) \in \mathcal{U}$ takes only discrete values. Denote by $|\mathcal{U}|$ the cardinality of the admissible control set. The optimal control sequence can be recursively computed by the following DP algorithm:

1) Initially, let $J_0(q_0) = 0$.
2) For k = 0,...,N-1, we have

$$J_{k+1}(q(k+1)) = \min_{q(k)}(L(q(k), u(k)) + J_k^*(q(k))),$$

where $q(k+1) = g(q(k), u(k))$.
3) Find the optimal control sequence associated with the optimal cost:

$$J^* = \min_{q(N)}(\psi(q(N)) + J_N(q(N))).$$

The advantages to using the DP algorithm lie in two areas. First, the DP algorithm does not require that the objective function $L(q(k), u(k))$ and system dynamics $g(q(k), u(k))$ be differentiable. Second, it guarantees finite convergence time, which is very important for real-time control applications. However, we have to admit that the "curse of dimensionality" of the DP algorithm may prevent the scalability of this approach.

VI. Simulation Results

The performance of the proposed NMPC based low-level vehicle control methods are demonstrated in three simulation examples. In all simulations, the distance l between the front and rear axles is 0.8. The steering angle ϕ lies in $[-\pi/4, +\pi/4]$. The saturation ranges of the control inputs v, ω are: $v \in [0, 5]$ and $\omega \in [-1, 1]$.

A. Free-space Way-point Navigation

In this scenario, a single vehicle moving in free space is considered. In the simulation, the reference trajectory is a SCC trajectory consisting of 7 segments. Starting at the origin $(0, 0)$, the reference linear velocity v_r is constantly equal to 1. Simulation results show that both approaches demonstrate excellent tracking performance (Figure 7).

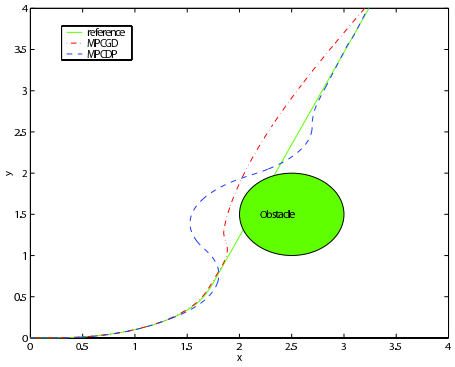

Fig. 8. Comparison of local obstacle avoidance for two MPC approaches

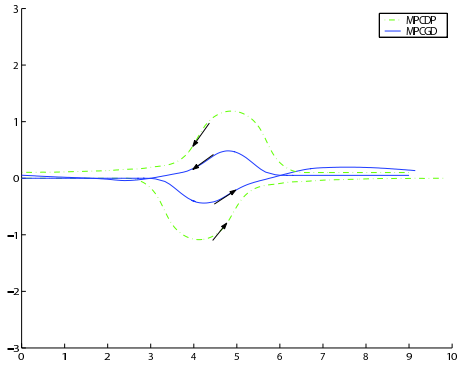

Fig. 9. Comparison of local collision avoidance for two MPC approaches

We compared the computation times for both approaches at each sampling period. The GD based approach has large variation, ranging from 0 to 1.5 seconds. The computation time variation of the DP based approach is pretty tight – about 0.02 seconds. Although the coding efficiency of *Matlab* may greatly affect the computation time, the large variation of computation cost may potentially present barriers to the application of the gradient descent based MPC approach in real-time control.

B. Trajectory Tracking With Obstacle Avoidance

In this simulation, we assume a circular obstacle is located along the reference trajectory with center at $(2.5, 1.5)$ and radius 0.5. Figure 8 shows that both approaches successfully avoid the obstacle. The vehicle trajectory with the DP based MPC approach has larger deviation from the reference trajectory than the vehicle trajectory with the GD based approach. The primary reason is due to the limited steering control capability.

C. Multiple Vehicle Tracking With Collision Avoidance

In this simulation, two vehicles have heading directions that are initially opposite to each other. We assume that each of the two vehicles is planning to move towards the other's location – they will collide in the middle of their ways. By adding the collision avoidance potential function component in the objective function, simulations show that both approaches yield good performance in avoiding collision (see Figure 9). The dashed curves are the vehicles' trajectories with the DP based MPC approach. Similarly as in previous simulations, the limited actuation capabilities result in large deviations from the reference trajectory compared with the GD based approach.

VII. Summary and Conclusions

In this paper, two problems in the lower level motion control of car-like UAVs are addressed: reference trajectory generation and constrained track following control. A SCC trajectory planner is investigated to provide a locally sub-optimal *reference trajectory* for car-like UAVs. To deal with practical constraints in the track following control design, as for example, local collision avoidance and actuator saturation, an MPC based approach is proposed to provide locally optimal control. Two approaches are proposed to solve the finite horizon optimal control problem: GD approach and DP approach. Both approaches have tradeoffs: the GD one provides adequate control accuracy, however, it suffers from convergence difficulties and large computation time variation; the DP approach sacrifices control accuracy to trade with stable computation time and robust performance.

References

[1] D. A. Schoenwald, "AUVs: In space, air, water, and on the ground," *IEEE Control Sys. Mag.*, vol. 20, no. 6, pp. 15–18, 2000.

[2] M. H. Douglas and A. Patricia, "Reducing swarming theory to practice for uav control," *Proc. IEEE Aerospace Conf.*, 2004.

[3] R. Olfati-Saber and R. M. Murray, "Distributed cooperative control of multiple vehicle formations using structural potential functions," in *Proc. 15th IFAC World Congress*, Barcelona, Spain, 2002.

[4] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[5] W. Xi, X. Tan, and J. S. Baras, "Gibbs sampler-based coordination of autonomous swarms," *Automatica*, vol. 42, no. 7, 2006.

[6] F. M. Y. J. Kanayama, Y. Kimura and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," *Proc. IEEE Int. Conf. Robotics and Automation, Cincinnatti, OH*, 1990.

[7] R. Fierro and F. Lewis, "Control of a nonholonomic mobile robot: Backstepping kinematics into dynamics," *J of Robotic Systems*, vol. 14, no. 3, pp. 149–163, 1997.

[8] E. Lefeber, J. Jakubiak, K. Tchon, and H. Nijmeijer, "Observer based kinematic controllers for a unycicle-type mobile robot," *Int. J. of Applied Math. and Comp. Sci.*, vol. 12, pp. 513–522, 2002.

[9] T. Lee, K. Song, C. Lee, and C. Teng, "Tracking control of unicycle-modeled mobile robots using a saturation feedback controller," *IEEE Tran. Cont. Sys. Techn.*, 9 (2), pp. 305–318, 2001.

[10] D. H. Shim, H. J. Kim, H. Chung, and S. Sastry, "A flight control system for aerial robots: Algorithms and experiments," in *Proc. 15th IFAC World Congress on Aut. Control*, July 2002.

[11] G. J. Sutton and R. R. Bitmead, "Computational implementation of NMPC to nonlinear submarine," *Chem. Process Cont.*, 26, 2000.

[12] J. C. Alexander and J. H. Maddocks, "Asymptotic stability and feedback stabilization," *Diff. Geometric Control Theory*, 1989.

[13] H. Sussmann, "The Markov-Dubins problem with angular acceleration control," *Proc. 36th IEEE CDC, San Diego, CA*, Dec. 1997.

[14] A. Scheuer and T. Fraichard, "Continuous curvature path planning for car-like vehicles," *Proceedings of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems, Grenoble, FR*, September 1997.

[15] X. N. Bui, P. Soueres, J. D. Boissonnat, and J. P. Laumond, "Shortest path synthesis for dubins nonholonomic robot," *Proc. of IEEE Int. Conf. on Robotics and Autom., San Diego*, May 1994.

[16] A. Scheuer, "Suboptimal continuous-curvature path planning for non-holonomic robots," *Proceedings of 11st Journes Jeunes Chercheurs en Robotique*, 1999, pp. 107–112.

[17] P. Morin and C. Samson, "Time-varying exponential stabilization of chained systems based on a backstepping technique," *Proc. of the IEEE Conf. on Decision and Control, Kobe, Japan*, 1996.

[18] J. S. Baras, X. Tan, and P. Hovareshti, "Decentralized control of autonomous vehicles," *Proc. 42nd IEEE Conference on Decision and Control*, vol. 2, Maui, Hawaii, 2003, pp. 1532–1537.