2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Receding Horizon Trajectory Optimization with a Finite-State Value Function Approximation

Bernard Mettler* and Zhaodan Kong

Abstract— **This paper describes a finite-horizon receding horizon trajectory optimization scheme which uses an approximation of the value function to provide cost-to-go (CTG) and associated state information. The value function approximation is computed using a finite-state, motion primitive automaton approximation of the vehicle dynamics. Using an actual value function approximation instead of heuristic CTG allows a tighter integration between the planning and control layers needed for vehicles operating in challenging spatial environments. It also enables a more rigorous use of the receding horizon control framework for autonomous control applications. The paper describes the finite-state value function approximation and its integration into the receding horizon scheme. Simulation examples illustrate the scheme's capabilities and highlight interesting open issues that will need to be addressed to take full advantage of the approach.**

I. Introduction

Spatial control of vehicle is fundamental to autonomous operation. Some of the most capable vehicle platforms, such as miniature rotorcraft, are highly dynamic systems. Aerial vehicles can move freely in three dimensional spaces and can exhibit a broad range of dynamic behaviors. To take full advantage of their capabilities without compromising safety, the spatial as well as vehicle state information must be accounted for in consistent manner. Essentially the planning and control must be integrated tightly, i.e., accounting for the vehicle's dynamics at the trajectory planning level and for the task or mission elements while controlling the vehicle.

Roboticists have been interested in the interaction of robots with their environments since the late 60's. A vast literature exists addressing various aspects of robot motion planning [13], [6], [14]. While trajectory planning can be rigorously formulated as an optimal control problem, solving them as such cannot be done sufficiently fast to enable the type of interactive capabilities. Therefore, for practical reasons, a hierarchic approach is often used: first obstacle-free path is determined (path planning), second a feasible trajectory is generated in a post-processing phase (robot control). Such decoupled techniques have been successfully applied to vehicles with relatively simple dynamics like ground robots. For vehicle with more complex dynamics, such as aerial vehicles, decoupling planning and controlling limits the performance and also makes it difficult to produce the type of performance or robustness guarantees needed for safe and effective operation. For vehicles with complex dynamics, operating in complex spatial environments and disturbed conditions, a more rigorous, control-theoretic approach is required.

Receding horizon (RH) optimization can help reduce the computational load by using a finite prediction horizon. Encouraging results have been obtained in simulations and experimental demonstration on a number of platforms [2], [19], [21], [17]. To take full advantage of the RH framework for motion planning, the key challenge is properly accounting for the discarded tail of the trajectory. In principle, like demonstrated for the control of nonlinear systems [18], [9], a RH scheme combined with an appropriate cost-to-go (CTG) function, can approach the performance of an infinite-horizon optimization.

Existing CTG methods for motion planning are mostly heuristic techniques based on geometric decompositions (like the visibility graph [2]) or cell decomposition [17]. These techniques do not explicitly account for the vehicle state and therefore result in ad-hoc implementation of the RH framework, making it difficult to attain high levels of control performance and to rigorously study the performance and robustness characteristics.

We can achieve a more rigorous implementation of RH trajectory optimization if the CTG function can be computed as an explicit approximation of the problem's value function (VF). In this paper we describe the integration of a RH trajectory optimization with an approximate value function computed using a finite-state approximation of the vehicle dynamics. Simulation results based on real vehicle data are used to illustrate the capabilities of the approach. We conclude the paper by outlining key issues that need to be addressed in order to make this approach a successful framework for spatial control of agile vehicles.

II. Background: Receding Horizon Trajectory Optimization

RH optimization is an attractive technique for controlling autonomous vehicles. It retains the strength of optimal control (accommodating nonlinear dynamics; explicitly accounting for hard constraints on state and control inputs; and using performance objectives), but requires a fraction of the computational load. RH optimization also provides a conceptual framework to rigorously divide the planning problem into an on- and offline problem [16]. The online RH optimization process focuses on the immediate environment and the most up-to-date information (from onboard sensors) while the offline (or near real-time) CTG computation process focuses on the more global, long-term elements. A rigorous understanding of the performance of RH schemes

*B. Mettler is an Assistant Professor at the Department of Aeronautical Engineering and Mechanics, University of Minnesota, Minneapolis, Minnesota, MN 55455 mettler@aem.umn.edu

Z. Kong is a graduate student at the same department.

will make it possible to precisely understand the fundamental relationship between computational load and control performance.

Truncating the optimization horizon can lead to stability and performance issues. Several techniques have been proposed to handle these [15]. The technique that is the most consistent with the original infinite-horizon form is to approximate the discarded tail of the optimization by a CTG function. For nonlinear control, it has been shown that Control Lyapunov Functions representing an upper bound to the true VF could be used as CTG function [9]. For trajectory planning amidst complex geographical environments this approximation cannot be used.

The main technique used so far for RH trajectory planning is based on Visibility Graph decompositions [2]. The graph is constructed based on the goal location and a polygonal obstacle configuration. CTG values for the graph's vertices can be computed using a shortest path algorithm. In a minimum-time problem the CTG is determined based on the edge length and an average vehicle speed. Maneuvering limitations can be accounted for heuristically. For example, a penalty based on the direction change between two edges at the vertices can be added to the cost. In [16], [17], a multi-resolution cell decomposition of the environment is used for CTG computation. Heuristics were used to incorporate vehicle state information: vehicle speed was related the the cell's dimension (via the minimum turn radius) and the optimal heading was assumed equal to the direction of the CTG gradient.

These existing CTG computation techniques have both practical and theoretical shortcomings. First, they make it very difficult to properly incorporate vehicle state information and account for vehicle dynamics. Therefore it is impossible to prescribe a terminal state (heading or velocity) at the goal; important control performance margins have to be used to ensure safety and robustness. Second, viewed from a control-theoretic standpoint, they are not explicit approximation of the VF of the underlying trajectory optimization problem. This makes it difficult to determine their precise performance and robustness characteristics. Empirical evaluations have to be used to determine appropriate implementation.

A. Trajectory Optimization Formulation

The task in a general trajectory optimization problem is to determine a control history $u(t)$ which drives the vehicle from its current state x_0 to a desired goal state x_{goal} while minimizing a chosen performance objective of the form:

$$J_\infty(\mathbf{x}) = \int_0^\infty g(\mathbf{x}, \mathbf{u}) dt, \tag{1}$$

where $\mathbf{g}$ is the instantaneous cost functional. The mathematical formulation is given as:

$$\begin{aligned} \min_{\mathbf{u}} \quad & J_\infty(\mathbf{x}) \\ \text{subject to} & \\ \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}) \\ \mathbf{x}(t=0) &= \mathbf{x}_0 \\ \mathbf{x}(t=\infty) &= \mathbf{x}_{goal} \\ \mathbf{u} &\in U(\mathbf{x}_v) \\ \mathbf{x}_p(t) &\in X_{free} \\ \mathbf{x}_v(t) &\in X_{vhc}(\mathbf{x}_v), \end{aligned} \tag{2}$$

where, $\mathbf{f}$ is the vector differential equation for the vehicles dynamics, and U is the set of admissible controls given the current state of the vehicle. The state vector $\mathbf{x}$ is partitioned in vehicle states $\mathbf{x}_v$ and vehicle position $\mathbf{x}_p$. X_{vhc} is the set of admissible vehicle states; X_{free} represents the admissible region of the environment (e.g. obstacle free).

The optimal command history $\mathbf{u}^*(t)$ at the current state $\mathbf{x}$ is given by:

$$\mathbf{u}^*_\infty(t) = \text{argmin}\{J^*_\infty(\mathbf{x})\}. \tag{3}$$

B. Receding Horizon Trajectory Optimization

In finite horizon RH optimization, the optimization horizon is truncated to a finite duration T. The problem is solved repeatedly to obtain the control action based on the most up-to-date state information. The horizon length T and the trajectory update time interval T_{updt} are typically set based on the online computational capabilities.

The optimal control for the RH+CTG scheme is the one that minimizes the composite cost

$$\mathbf{u}^*_T = \text{argmin}\{J_T(\mathbf{x}(\mathbf{t})) + J_{CTG}(\mathbf{x}(t+T))\}, \tag{4}$$

where $J_T(\mathbf{x}(\mathbf{t}))$ is the finite-horizon cost

$$J_T(\mathbf{x}(\mathbf{t})) = \int_0^T g(\mathbf{x}(\mathbf{t}), \mathbf{u}) d\tau, \tag{5}$$

and $J_{CTG}(\mathbf{x}(t+T)$ represents the cost of the discarded tail of the trajectory or *cost to go*. It is a function of the vehicle state attained at horizon end $\mathbf{x}(\mathbf{T})$.

Note that if $J_{CTG}(\mathbf{x})$ is identical to the optimal, infinite-horizon cost $J^*_\infty(\mathbf{x})$, the RH- and the infinite-horizon solutions are identical, i.e., the optimality gap is zero. $J^*_\infty(\mathbf{x})$ is also called the optimal value function (VF) $V^*(\mathbf{x})$ [20]. Hence, ideally, the CTG should be chosen as an approximation of the VF $\tilde{V}^*$.

C. Cost-To-Go for Trajectory Optimization

The optimal value function is the solution to the Hamilton-Bellman-Jacobi (HBJ) differential equation [20], [5]. However, for many problems of practical interest, this equation cannot be solved analytically and computational techniques have to be used. A brute-force approach would be to numerically solve the infinite-horizon optimization problem over a spatial grid $\mathbf{x}_{pos} \in X_{free}$.

Since the VF is a function of the environment and the goal state. To accommodate changes occurring when operating in partially known or dynamically changing environments, it is critical to develop computationally efficient VF algorithms. We also need to understanding what approximation level in the VF would be sufficient to enable the RH+CTG scheme to perform at a specified performance level.

III. Finite-State Value Function Approximation

In the following we formulate the CTG computation as a finite-state approximation of the continuous optimal control problem. Such a motion primitive automaton (MPA) representation has originally been proposed as part of a hybrid guidance system [7]. With a finite-state representation of the vehicle dynamics, the trajectory optimization becomes a sequential decision problem which can be solved as a dynamic program [3].

A. Hybrid Guidance vs. RH-CTG

The finite-state MPA is obtained from a quantization of the system dynamics [10]. In contrast to the hybrid guidance scheme [7], when used as CTG function in the RH+CTG scheme, the finite set of states and control does not directly constrain the vehicle control and the resulting trajectories. This is because the online optimization uses a model of the actual dynamics of the system; the choice of finite states affect the control performance and behavior only in so far that they constrain the system's states at the horizon end, and indirectly through their effect on the CTG accuracy. Therefore, much simpler MPA can be used in a RH-CTG scheme addressing a potential issue due to the *curse of dimensionality* [3].

B. RH-CTG Formulation

The receding horizon optimization with the finite-state value function can be formulated as:

$$\mathbf{u}_T^* = \text{argmin}\{J_T(\mathbf{x}(\mathbf{t})) + \tilde{V}^*(\mathbf{x}(t+T))\}. \quad (6)$$

The additional constraint is that at the horizon end, the system state must be equal to the finite state associated with the finite-state value function $\tilde{V}^*(\tilde{\mathbf{x}})$.

$$\mathbf{x}(t+T) = \tilde{\mathbf{x}}. \quad (7)$$

1) RH-CTG Design Problem Statement: The idea is that with an appropriate finite-state model and RH implementation (selection of prediction model, trajectory update rate, sampling time, horizon length) the optimality gap due to the finite-state approximation can be partially recovered. Therefore the RH-CTG control design problem can be stated as: find the lowest dimensional finite-state MPA and the RH implementation that satisfies the performance specifications. The required VF update is determined by the dynamics of the environment (and task). The practical constraints for this design problem are the online computational resources (processing and storage).

One way to measure the quality of the finite-state approximation is through the optimality gap between the actual value function and its approximation $\Delta V^*(x) = \tilde{V}(x) - V(x)^*$. However, ultimately the form of implementation of the RH optimization scheme plays an important role in determining what the optimality gap will be under the RH+CTG scheme.

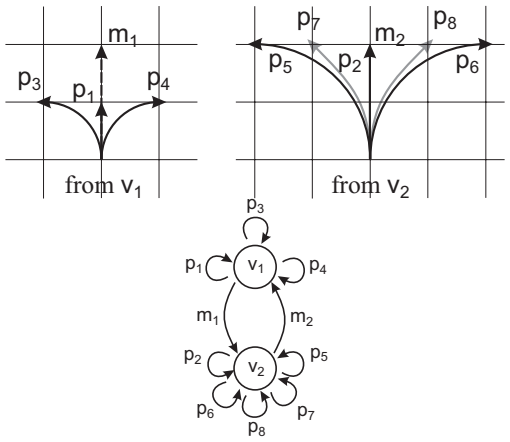

Fig. 1. (top) Illustration of an example motion primitive automaton (MPA). The motion primitives (MPs) are specified for quantized vehicle speeds. Transition between the speeds is enabled by acceleration and deceleration primitives. Each MPs trajectory segments start and end on a grid. (bottom) The transition graph of the MPA.

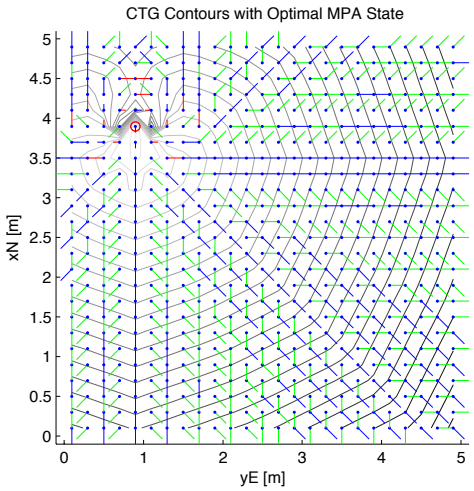

Fig. 2. Finite-state value function map computed based on the motion primitives illustrated in Figure 1. The field corresponds to a 4x3 meter region we setup to perform our evaluation. The plot shows the contour lines of the CTG values (spacing 0.5 sec) together with the vector field describing the optimal MPA state at each grid point (shown by dot). The line length indicate the velocity and the color indicate the type of MP (blue: straight p_1, p_2 or m_1, m_2; green: $\pi/4$ turns p_7, p_8; red: $\pi/2$ turns $p_3 \dots p_6$). (bottom) Trajectories from a receding horizon planner.

C. Finite-State Vehicle Model

As a starting point we use a simple planar MPA. The preliminary results will help us illustrate the approach and highlight the overall performance issues.

1) Basic MPA: The state-space for the basic 2D trajectory planning problem consists of the: position in a geographical reference system $\mathbf{x}_p = [x_N, y_E]^T$ the vehicle heading ψ and vehicle longitudinal speed u. The MPA are typically obtained by quantizing the vehicle dynamics. In the following we

perform a quantization based on specified spatial grid. The state-space is discretized at a linear resolution d_{xy} (set to the minimum turn radius for the the lowest, non-zero speed) and heading resolution of $d_\psi = \pi/4$. Using motion primitives (MPs) defined on a fixed spatial grid prevents from having to perform costly interpolations during the value iteration. The velocity is discretized based on practical considerations.

The MPs must start and end on a grid point and their initial and final heading must be a multiple of $\pi/4$. Two forms of MPs are used: rectilinear or turns. Each can be steady-state or accelerating/decelerating. MPs are described by the following attributes: translational and angular displacements (with respect to the discretized coordinates and heading); the incremental cost (travel time for a minimum time performance objective); and the connectivity information. These attributes are determined based on the performance of the tracking control system.

Figure 1 (top) depicts the MPs used in our example. Only two speed levels are used ($V = \{v_1, v_2\}$) are used. The motion primitives include for each speed: a straight cruise (p_1 and p_2); a left and right $\pi/2$ turn ($p_{4..6}$); and left and right $\pi/4$ turn ($p_{7..10}$). Also included are one straight acceleration and one deceleration primitive (m_1 and m_2) needed to transition between v_1 and v_2. The set of decision variables for the MPA at each speed are $\mathbf{P}_{v_1} = \{p_1, p_3, p_4, m_1\}$ and $\mathbf{P}_{v_2} = \{p_2, p_5, p_6, , p_7, p_8, m_2\}$. Figure 1 (bottom) shows the state transition graph of the MPA.

The example is based on the Blade CX helicopter and control system currently implemented on our interactive flight test facility [11], [12]. The controller enables tracking accelerating velocity references of up to $a_{max} = 2.5m/sec^2$. The top velocity is selected to be $v_2 = 1$m/sec and low velocity v_1 is selected to be 0.4m/sec.

2) CTG computation: The CTG is computed using a value iteration implemented as a label correcting approach [4]. Figure 2 shows the finite-state value function computed based on the motion primitives illustrated in Figure 1 for an obstacle free environment. The cost-to-go is shown as contour lines (intervals correspond to 0.5 sec). The line segments illustrate the vector field based on the optimal MPA states (grid points shown by dot). The MP state p is given by the color (see caption). The VF computation for this field using Matlab is 0.985 sec (AMD Athlon 64X2 Dual Core Processor 5000+ 3.2GHZ).

D. Integration with RH Trajectory Optimization

Figure 3 illustrates the planning process and Figure 4 illustrates the blockdiagram of the planner system. At each trajectory update time T_{updt}:

1) a set of local VF points is extracted from the VF map based on the current system state
2) the set is pruned using heuristics to reduce candidate CTG points
3) the remaining set of points $\Upsilon_{\hat{\mathbf{x}}} = \{\tilde{V}^*, \tilde{u}^*, \tilde{\psi}^*\}$ are used as terminal constraints for the online trajectory planner
4) CTG point and associated trajectory that achieves the minimal composite cost is selected

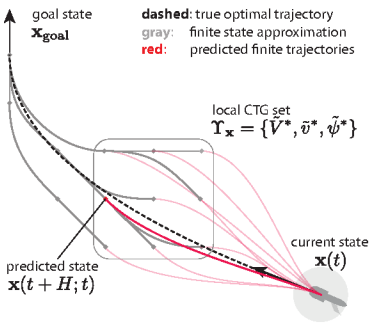

Fig. 3. Illustration of the RH+CTG optimization process.

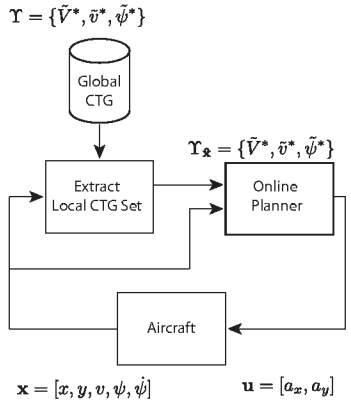

Fig. 4. Illustration of the receding horizon trajectory planning scheme with an approximate finite-state value function.

5) the control input solution for the first time step is implemented $\mathbf{u} = [a_{lon}, a_{lat}]$

1) Extraction of local VF points: In principe the local VF set should be chosen as the set of points that are reachable from the current state under the finite RH planner:

$$\Upsilon_{\hat{\mathbf{x}}} = R(\mathbf{x}(t), T). \tag{8}$$

In our implementation we approximate the reachable set based on the velocity and heading. Furthermore, to reduce the computational load we prune the set based on a heuristic. A more efficient techniques would be to generate an analytic representation of the local VF map.

2) Online RH trajectory optimization: A variety of optimization tools can be used for the online planner. Mixed integer linear program (MILP) has been used extensively. In our current implementation we use a nonlinear programing (NLP) technique based on interior point method [1].

The vehicle dynamics are modeled as a second-order discrete-time system. This model describes the closed-loop dynamics of the vehicle under a the velocity tracking controller.

Figure 5 shows one trajectory implemented in the obstacle free field. The parameters for the RH implementation are: trajectory update rate $T_{updt} = 0.1$sec; sampling rate of the

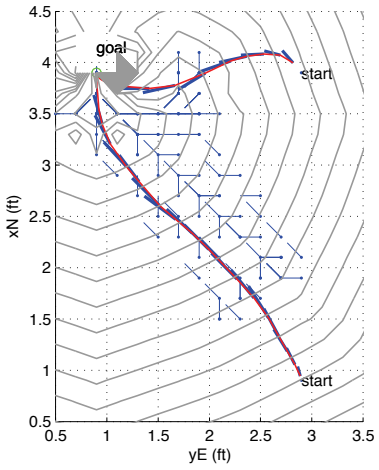

Fig. 5. Trajectory (in red) obtained from the RH+VF scheme with the VF shown in Figure 2 for a goal $\mathbf{x}_{goal} = [4, 0]^T$, and final states $\psi_{goal} = 0$ and velocity $u_{goal} = v_1 = 0.4$m/sec. Also shown are the CTG points with associated states that were extracted along the way.

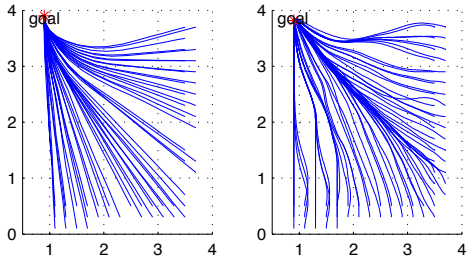

Fig. 6. Comparison of the trajectories obtained using the NLP planner (left) and those obtained with the RH+CTG planner (right) using the VF illustrated in Figure 2.

discrete time model used in the planner $T_s = 0.2$sec; 5 evenly space nodes for the prediction horizon. The constraints for the online planner are $a_{max} = 2.5m/sec^2$ and $u_{max} = 2m/sec$. The local VF set is extracted based on the vehicle heading (33 local VF points are extracted from a triangular region in front of the vehicle; the set is pruned by keeping the five points with lowest CTG values).

IV. Performance Evaluation

To evaluate the overall behavior of the of the RH+CTG scheme, its computational performance, and its ability to recover the optimality gap (resulting from the MPA-based finite-state VF approximation) we used a simple, planar goal-directed control task. The scale of the problem is chosen to enable to fully exercise the maneuvering capabilities of the 200-gram Blade-CX helicopter used in our facility.

The goal is located at $\mathbf{x}_{goal} = [4, 0]^T$, and the final state is $\psi_{goal} = 0$ and velocity $u_{goal} = v_1 = 0.4$m/sec. At this point in time, all results are based on simulations using a model of the helicopter.

A. Performance Benchmark

As benchmark for optimality we formulated a nonlinear program (NLP) that we solved using an interior point method (IPOPT) [22]. We chose a minimal parameterization (20 evenly space nodes), just sufficient to achieve good, full-horizon performance for all start-goal configurations in our test task. The NLP trajectories that are used are open loop (they were computed once at the starting points). This way we can consider these trajectories as the true optimal ones for the system.

Figure 6 shows the trajectories resulting from the RH scheme and the corresponding NLP ones. The RH trajectories display more variability that the open-loop NLP ones but are qualitatively quite similar to the NP ones. There is a quite a strong bias toward the fields diagonal which is likely due to the coarse heading discretization in the MPA.

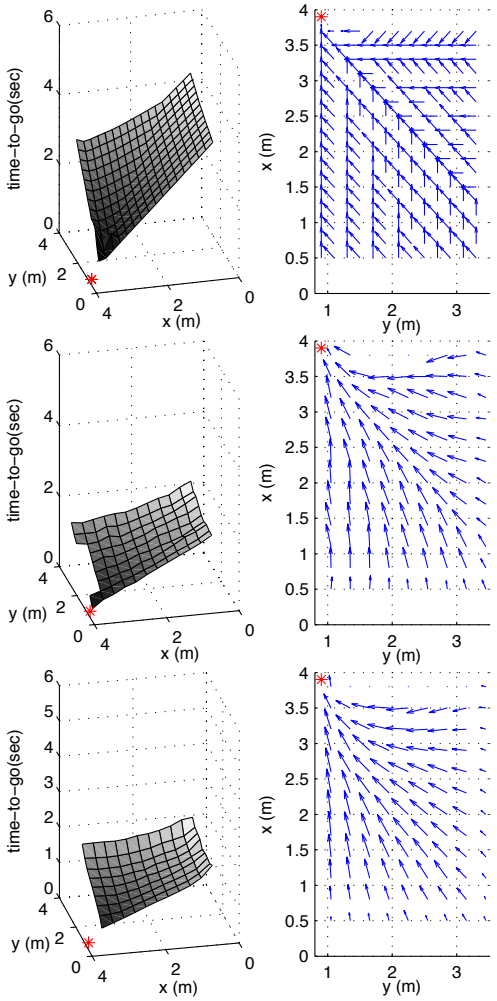

Fig. 7. Performance comparison of the control schemes: (top) only finite-state system; (middle) RH planner with cost-to-go; (bottom) nonlinear trajectory optimization with full horizon.

B. Vector Field Comparison

To better compare the performance we computed the $x-y$ spatial distributions of the actual cost-to-go and vehicle states as vector field . The values were computed by averaging the cost and system states using a cell decomposition ($d_{xy} = 0.3$ m) of the test field. Figure 7 shows the comparison between the finite-state approximation and the two online planners:

(top) finite-state system; (middle) RH planner with cost-to-go; (bottom) nonlinear trajectory optimization. First, notice that the RH scheme recovers a significant portion of the the optimality gap (time-to-go for the RH and NLP plots) and performs close to the full-horizon nonlinear planner. This is achieved by taking advantage of the full range of speed (up to 2m/sec) and the continuous control inputs and states.

C. Computational Load

Finally, we evaluated the computational load for the two planning schemes. Table I gives the statistics of the planner performance for our test task. The trajectory performance is measured by the optimality gap with respect to the full-horizon, NLP planner; the computational efficiency of the planner is measured by the average CPU time per trajectory update (based on the AMD Athlon processor). All the values were obtained by averaging over all the trajectories of our test task.

We see that the finite-state approximation using the MPA has an average gap of 1.34 sec (also visible in Figure 7). This is quite large considering that the total trajectory duration is of the order of 1.5 to 2.5 seconds. We also see that the average optimality gap is near zero for the RH scheme. The computing times are on average at least half for the RH scheme. Also, the RH optimization has a much smaller standard deviation. The NLP can take vastly different computing times from one case to the next.

TABLE I

AVERAGE OPTIMALITY GAP AND COMPUTATIONAL PERFORMANCE.

	ΔJ	$\sigma_{\Delta J}$	T_{CPU}	$\sigma_{T_{CPU}}$
MPA	1.34	0.38	0	0
RH+CTG	0.043	0.22	.68	0.55
NLP	0	0	3.84	3.13

When interpreting these results it is important to realize that the RH+CTG scheme's computational load is independent of the planning problem, while the full-horizon NLP's parameterization will have to be increased for larger problems. For example, increasing the horizon length to accommodate larger problems is bounded above by the exponential scale and is NP-hard with respect to the number of nodes [8]. Finally, the RH+CTG performance can still be improved significantly by using better heuristics to extract the smallest local CTG set or alternatively use local point to generate an analytic representation of the local VF. There is also significant improvement that can be achieved using more efficient software implementations (currently based on Matlab and Simulink).

D. Application to Obstacle Field

The RH control scheme and VF computation can be readily applied to a terrain environments. Figure 8 illustrate the results for a fictional environment setting in our indoor lab space with three obstacles. It shows the resulting trajectories for three different starting points in the obstacle field. Also shown are the local VF points used by the planner as it progresses through the environment. The goal coordinates are $\mathbf{x}_{goal} = [4.2, 2]'$ with same goal state as previously. The environment is provided to the VF value iteration algorithm as an occupancy grid. Any point which is inside an obstacle region is assigned an infinite cost-to-go value.

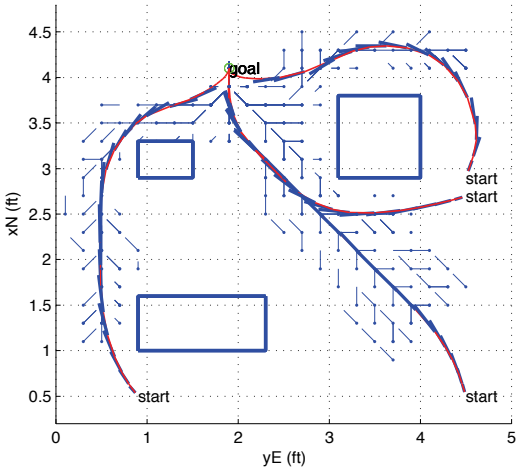

Fig. 8. Trajectories for the RH-CTG scheme in an obstacle field. The plot also shows the local VF points that were retrieved along the way.

Figure 9 illustrates the importance of taking into account the state in addition to the cost-to-go values in RH schemes. The trajectories were obtained by using the same starting state and goal location, but four different goal states ($\psi_{goal} = 0, \pi/2, \pi, -\pi/2$ and $u_{goal} = v_1$). Notice how these differences result in large differences in the trajectories.

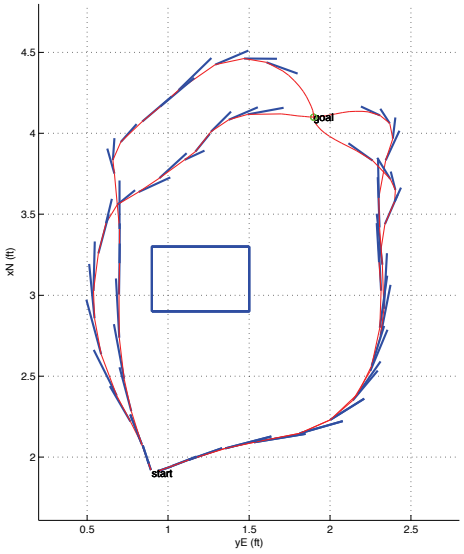

Fig. 9. Trajectories for the RH-CTG scheme in an obstacle field. All trajectories have the same starting state but each a different goal states are used (terminal heading $\psi_{goal} = 0, \pi/2, \pi, -\pi/2$).

V. CONCLUSIONS

The results obtained in this paper are encouraging but they also highlight important questions that will need to be answered if we want to take advantage of the RH+CTG schemes full potential for autonomous vehicle control.

A. Highlights and Summary

The main results of this initial study are:

- A relatively simple MPA model of the vehicle can be used to approximate the VF. By itself, the VF map already conveys information about the relationship between the spatial environment and the vehicle behavior.
- VF map provides a control-theoretic framework to study the complex interaction between a vehicle and the environment (spatial and behavioral dimensions). It should provides a way to explicitly relate the algorithmic quantities to vehicle dynamics capabilities (e.g. close-loop control requirements), the environment characteristics, and task requirements.
- Combined with online receding horizon control, the VF map should enable computation of a trajectory in real time which approaches the performance of the true optimal trajectory, but is computationally cheaper.
- The finite-state VF computation does not require any geometrical processing of the environment data. Environments given in form of digital elevation maps or occupancy maps can be used directly in this scheme.
- A variety of optimization techniques can be used in the RH planner. This makes it possible to apply this scheme to a variety of nonlinear systems.
- We expect that like in other application, the RH schemes should display good robustness in the face of uncertainties and disturbances.

B. Current and future work

These results also raise several important questions. First, we need to better understand the relationship between the level of approximation in the MPA, its effect on the VF accuracy, and on the performance of the RH+CTG scheme. This will help us develop methods and guidelines to design a system that achieves the best performance given the available on-board computational resources, the characteristics of the environment and operational conditions, and the capabilities of the vehicle.

In general, the behavior of the RH+CTG schemes for nonlinear systems as a function of implementation settings (e.g. length of optimization horizon and trajectory update rate) is hard to predict. Previous results were mainly based on Control Lyapunov functions (CLF) [18]. For the special case of vehicular planning tasks and by using the finite-state VF approximations we expect to be able to obtain stronger results.

Most real-world problems are characterized by uncertain, partially known environments, and disturbed operating conditions. We are extending the CTG computation and RH implementation to account for these stochastic effects. This will enable analyzing the robustness and performance characteristics of the system and develop the type of performance and robustness guarantees needed for autonomous control systems. Finally, we are in the process of implementing this scheme in our interactive flight research lab.

REFERENCES

[1] Andreas Wachter adn Lorenz Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25, 2006.

[2] J. Bellingham, A. Richards, and J. How. Receding Horizon Control of Autonomous Aerial Vehicles. volume 5, pages 3741–3746. American Control Conference, May 2002.

[3] R. Bellman and S.E. Dreyfus. *Applied Dynamic Programming*. Princeton University Press, Princeton, New Jersey, 1962.

[4] D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, Belmont, MA, 1995.

[5] A. E. Bryson and Yu-Chi Ho. *Applied Optimal Control*. Taylor and Francis, Bristol, PA, 1975.

[6] H. Choset, K.L. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion; Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, 2005.

[7] E. Frazzoli, M.A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.

[8] P. Grieder and M. Morari. Complexity reduction of receding horizon control. Proceedings. 42nd IEEE Conference on Decision and Control, Dec. 2003.

[9] A. Jadbabaie, J. Yu, and J. Hauser. "Unconstrained Receding-Horizon Control of Nonlinear Systems". *IEEE Transactions on Automatic Control*, pages 776–783, May 2001.

[10] Donald E. Kirk. *Optimal Control Theory*. Dover Publications, Inc., Mineola, NY, 1998.

[11] N. Kundak and B. Mettler. Experimental framework for evaluating autonomous guidance and control algorithms for agile aerial vehicles. Kos, Greece, July 2007. Proceedings of the European Control Conference (ECC).

[12] N. Kundak, M. Rhinehart, and B. Mettler. Modeling and control design for miniature autonomous coaxial rotorcraft. Montreal, Canada, April-May 2008. Proceedings of the 64th Forum of the American Helicopter Society for Forum of the American Helicopter Society.

[13] J.C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, 1991.

[14] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[15] D.Q. Mayne, J.B. Rawling, C.V. Rao, and P.O.M. Scokaert. "Constrained model predictive control: Stability and optimality". *Automatica*, 36(6):789–814, 1987.

[16] B. Mettler and E. Bachelder. "Combining On- and Offline Optimization Techniques for Efficient Autonomous Vehicle's Trajectory Planning". Number AIAA 2005-5861, San Francisco, CA, August 2005. AIAA Guidance, Navigation and Control Conference and Exhibit.

[17] B. Mettler and O. Toupet. Receding horizon trajectory planning with an environment based cost-to-go function. Seville, Spain, Dec. 2005. Proceedings of the joint ECC-CDC Conference.

[18] James A. Primbs, Vesna Nevistic, and John C. Doyle. "A Receding Horizon Generalization of Pointwise Min-Norm Controllers". *IEEE Transactions on Automatic Control*, 45(5):898–909, May 2000.

[19] A. Richards, Y. Kuwata, and J.P. How. Experimental Demonstration of Real-Time MILP Control. Austin, TX, Aug. 2003. AIAA Conference on Guidance Navigation and Control.

[20] Eduardo D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems.* Springer, New York, NY, 1998.

[21] M. Valenti, T. Schouwenaars, Y. Kuwata, E. Feron, J. P. How, and J. Paunicka. Implementation of a manned vehicle - uav mission system. Number AIAA-2004-5142. AIAA Guidance, Navigation, and Control Conference, Aug 2004.

[22] A. Wachter and L.T. Biegler. On the implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming. Technical report, IBM T. J. Watson Research Center, USA, March 2004.