Proceedings of the 38th
Conference on Decision & Control
Phoenix, Arizona USA • December 1999

A Hybrid Control Architecture for Aggressive Maneuvering of Autonomous Helicopters

Emilio Frazzoli [1] Munther A. Dahleh [2] Eric Feron [3]

Abstract

In this paper we propose a hierarchical control architecture for aggressive maneuvering applicable to autonomous helicopters. In order to reduce the computational requirements of the control problem to be solved to achieve aggressive trajectories, a hybrid system framework is used, which allows for a substantial reduction in the complexity of the system, as well as for guarantees on the stability of the overall behavior. The hybrid controller is based on an automaton whose states represent feasible trajectory primitives. The selection of maneuvers, and hence the generation of the complete trajectory, can be cast as an optimal control problem that can be solved efficiently in real time.

1 Introduction

Several envisioned mission scenarios for unmanned aerial vehicles (UAVs) require aggressive flying in an unknown, dynamic and potentially hostile environment. The adjective *aggressive*, when referred to flight control, stands for "more severe, intensive, or comprehensive than usual" and is often used when addressing maneuvers performed at the edge of the flight envelope, that is maneuvers that push the limits of the aircraft performance . The ability of the UAV to use its maneuvering capabilities to the fullest, as well as the ability to react in real time to changes in the operational environment is obviously of paramount importance. On manned aircraft, maneuvers are initiated by a human pilot, and the flight control system is called upon to avoid departure from controlled flight, or violation of the flight envelope constraints. However, when dealing with fully autonomous vehicles, the word aggressive acquires a broader meaning: not only we desire to perform such maneuvers, but we need to appropriately select and initiate them, in order to meet some higher level objective. With this we mean that in general we are not interested in executing an aggressive maneuver for its own sake (except, for example, for demonstration purposes), but because the maneuver will place the vehicle in a more favorable position to reach a predetermined and meaningful goal (such as moving to a certain location, evading a threat, and so on). The "aggressiveness" of such a maneuver planning can also be seen as having no (or little) concerns about the use of the vehicle resources, such as the control effort. Finally, we note that aggressiveness in general will imply the temporary departure from what are usually considered "stable" operating conditions, eventually allowing for a "recovery" phase at the end of the maneuvering flight. Having all of the above considerations in mind a natural description of aggressive maneuvering can be translated, in many cases of interest, into a **minimum time** optimal control problem, within the constraints of the vehicle capabilities in the full flight envelope: this is the definition of "aggressive" that we will use in the following.

In general, the solution to optimal control problems is prohibitive from the computational point of view, especially when it is desirable to fully exploit the performance characteristics of the vehicle (for a review of trajectory optimization techniques, see [1]). In order to reduce the computational complexity of the problem, without sacrificing too much of the vehicle capabilities, we restrict the class of nominal trajectories to the family of trajectories that can be generated by the interconnection of appropriately defined primitives. These primitives will then constitute a "maneuver library" from which the nominal trajectory will be constructed. Instead of solving an optimal control problem over a high-dimensional, continuous space, we will solve a mixed integer programming problem, over a much smaller space. The idea expressed above translates into an integrated hierarchical control architecture applicable to autonomous aerial vehicles (details will be given for application to small helicopters). At the core of the control architecture lies a hybrid automaton, the states of which represent feasible trajectory primitives for the vehicle. Each constituent subsystem of the hybrid controller will be the agent responsible for the maneuver execution. The task of the automaton will be the generation of complete, feasible and "optimal" trajectories, via the interconnection of the available primitives. Apart from the reduction in computational complexity, one of the objectives of this approach is the ability to provide a mathematical foundation for generating a provably nominally stable hierarchical system, and eventually develop the tools to analyze robustness in the presence of uncertainty in the process as well as in the environment.

2 Related work

The design of flight control systems for autonomous vehicles has been the object of a relevant body of research in the recent past: one of the reasons can be seen in the dramatic increase in the computational power available on

[1]Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, email:frazzoli@mit.edu

[2]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, email: dahleh@lids.mit.edu

[3]Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, email: feron@mit.edu

small autonomous vehicles. A hierarchical decomposition of the tasks required for an autonomous vehicle control is usually performed when dealing with *intelligent* flight control for autonomous vehicles (see for example [2] and references therein). The usual decomposition includes a **guidance** layer for trajectory planning, and a **control** layer for command tracking. On top of both layers, we can define a **mission planning** level, that defines the current objective. The different levels are usually interacting only on a very limited basis, and most often a formal analysis of the complete system is very difficult, or based on time-scale separation arguments, that could be inapplicable to the control of vehicles in a dynamic environment. The guidance function is usually composed by a flight planner that generates a trajectory according to a kinematic model of the vehicle, often disregarding, or severely limiting, its actual dynamics. The generated trajectory will be in general infeasible for the vehicle: this can be obviated if the trajectories do not require "aggressive" flying, but can lead to very conservative behavior. Moreover, in many cases where the trajectory planner builds a sequence of flight modes, the selection is carried out by procedural or rule-based algorithms which, again, can be very difficult to analyze [3, 4, 5, 6]. Several authors have concentrated on the maneuver tracking aspect in the general framework of nonlinear control: to name just a few of the papers published in the recent past, we can mention [7, 8, 9, 10, 11]. In all these papers powerful techniques are developed to track a given reference command, but the question of generating an optimal trajectory remains open.

3 Hierarchical decomposition

The architecture that we will adopt in this work is based on the following decomposition . At the highest level, we have a **strategic** control layer, in which the current mission objectives are defined. Mission planning and way-point generation are typically included at this level. In a manned mission, this can be identified with the kind of information that is provided to pilots during pre-flight briefing, and possibly by operational scenarios updates. An intermediate layer can be defined as the **tactical** level. This can be seen as the level at which guidance and trajectory planning are usually included at this level. We argue that a human pilot at this level of abstraction decides the best sequence of "maneuvers", or trajectory primitives, that ensures the fulfillment of the mission objectives. At the lowest level, we have the actual interaction with the physical plant: this is sometimes referred to as **skill** or **reflexive** level, and includes the traditional control functions (stabilization, regulation, command tracking). Skilled human pilots are usually performing these functions without a conscious effort, at least if the vehicle performance provides good handling qualities: hence the name of reflexive level.

The core of the architecture we propose is represented by a hybrid automaton, which roughly performs the functions associated with the tactical level described above, that is, it selects the optimal maneuver to be executed in order to minimize a suitable cost function, provided by the strategic layer, given the current state and within the constraints of the vehicle dynamics. Each constituent subsystem of the hybrid controller will be the agent responsible for the maneuver execution; as stated in the introduction, the task of the automaton will then be the generation of a complete, "optimal", trajectory, via the concatenation of the available primitives.

4 Trajectory Primitives

We will restrict the nominal trajectories to the family of curves that can be generated by the interconnection of appropriately defined primitives. We want to characterize trajectory primitives in order to: (1) capture the relevant characteristics of the vehicle dynamics; (2) allow for the creation of complex behaviors from the interconnection of primitives (we want to obtain "good" approximations to optimal solutions) (3) identify the *minimal* set of key parameters to be exchanged between hierarchical levels; this is even more important for extension to multi-vehicle operations, or more complex systems. A brief exposition of the helicopter dynamics will be given in the following, with the purpose of leading and motivating our choice of trajectory primitives.

4.1 Helicopter Dynamics Model

In this section, we will briefly introduce the helicopter dynamics model that is the basis of the control architecture design. The discussion of some fundamental properties of the helicopter dynamics will clarify and motivate the specialization of the hybrid control system structure that will be presented in the later sections. We need a model that is powerful enough to adequately represent the helicopter dynamics characteristics over the full flight envelope, but that at the same time can be used for real-time trajectory planning and execution: this means that we will have to sacrifice part of the accuracy obtainable only through very detailed and complicated models. Such a model can be derived by physical intuition and first principles: at the basis of the model we have the rigid body dynamics, coupled with momentum theory and basic aerodynamics for the computation of the forces acting on the helicopter (see for example [12]). Similar minimum complexity models have been recently used for helicopter flight control design [13, 10] and system identification [14]. We start by stating the equation of motion for the helicopter as a rigid body [15]. The configuration of the helicopter will be described by an element g of the Special Euclidean group in the three-dimensional space, usually denoted by $SE(3)$. Using homogeneous coordinates, a matrix representation of $g \in SE(3)$ is the following:

$$g = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \qquad (1)$$

where $R \in SO(3)$ is a rotation matrix and $p \in \mathbf{R}^3$ is a translation vector. The kinematics of the rigid body are determined by $\dot{g} = g\hat{\xi}$ where $\hat{\xi}$, usually denoted as *twist*, is an element of the Lie algebra $\mathbf{se}(3)$ associated with $SE(3)$. A matrix representation of an element $\hat{\xi} \in \mathbf{se}(3)$ is

$$\hat{\xi} = \begin{bmatrix} \hat{\omega} & v \\ 0 & 0 \end{bmatrix} \tag{2}$$

where ω and v are respectively the angular and translational velocities in body axes, and the skew matrix $\hat{\omega}$ is the unique matrix such that $\hat{\omega} u = \omega \times u$, for all $u \in \mathbf{R}^3$ The dynamics equations, in matrix notation, will be given by:

$$\begin{aligned} J_b \dot{\omega} &= -\omega \times J_b \omega + M_b(g, \xi, u) && (3) \\ m\dot{v} &= -\omega \times mv + F_b(g, \xi, u) && (4) \end{aligned}$$

where M_b and F_b represent the torques and forces in body axes, which are in general a function of the vehicle state, and of the control inputs u. A simplified expression for the body wrench, commonly used in minimum-complexity models, can be given as:

$$\begin{aligned} F_b &= R^T w + F_{main}(u) + F_{tail}(u) + F_{aero}(\xi) && (5) \\ M_b &= r_{main} \times F_{main}(u) + r_{tail} \times F_{tail}(u) + \\ & \quad + M_{rotor}(F_{main}, v) + M_{aero}(\xi) && (6) \end{aligned}$$

$$\begin{aligned} F_{main} &= T_m[-a_1, \, b_1, \, -1]^T && (7) \\ F_{tail} &= F_t[0, \, 1, \, 0]^T && (8) \end{aligned}$$

where w is the gravitational force (weight), F_{aero} and M_{aero} represent the aerodynamic wrench, M_{rotor} is the reaction torque from the main rotor, r_{main} and r_{tail} are the relative positions of the main and tail rotor hubs with respect to the center of mass. Finally, we have the four (pseudo)-controls, namely the pitch and roll flapping angles a_1 and b_1, giving the orientation of the main rotor no-feathering plane in body axes, the main rotor thrust T_m, and finally the side force generated by the tail rotor and vertical stabilizer.

4.2 Symmetry

Under fairly reasonable assumptions, such as homogeneous and isotropic atmosphere, and constant gravity acceleration, the dynamics of a flying vehicle are invariant to translation and rotation about a vertical axis. The subgroup $H = \mathbf{R}^3 \times SO(1) \subset SE(3)$, where the rotation is only performed about the vertical axis, is hence a symmetry group for the vehicle dynamics, that is if $h \in H$ then it follows that $\dot{\xi}(g, \xi, u) = \dot{\xi}(gh, \xi, u)$. An element $h \in H$ is completely described by the translation vector $p \in \mathbf{R}^3$ and the heading angle $\psi \in [0; 2\pi)$. A key consequence of the symmetry of the dynamics is that we can treat all trajectory primitives as equivalence classes, and choose a prototype for each primitive, starting at a reference position and heading (say at the origin, heading north).

4.3 Trim trajectories

As a first class of primitives, we will consider trim trajectories. These are defined as those trajectories along which the velocities in body axes (the twist) and the control input are constant. From the above discussion of the symmetry properties, all trim trajectories will be the composition of a constant rotation g_0 and a screw motion $h(t) \in H$, given by the exponential of an element η of the Lie sub-algebra $\mathbf{h} \subset \mathbf{se}(3)$. This screw motion can be visualized in the physical space by a helix flown at a constant sideslip angle. Such helices are usually described by the following parameters: V, the magnitude of the velocity vector; γ, the flight path angle; $\dot{\psi}$, the turning rate; and finally β, the sideslip angle. The trim trajectory will then be completely described by quantities $\bar{g}$, $\bar{u}$, and η (or equivalently, the parameter vector $\bar{T} := \{V, \gamma, \dot{\psi}, \beta\}$). It should be mentioned that for a given choice of η, several choices of $\bar{g}$ and $\bar{u}$ are possible, and the selection of desirable values for them is the outcome of some (off-line) design process. Note that for fixed wing aircraft, β is usually assumed to be zero, or very small (coordinated flight). This is not necessarily true for helicopters, especially for low velocity regimes (e.g. sidesteps and backwards motion can be accomplished by helicopters). The first step in the design of our control architecture is the selection of a number of trim trajectories. The selection of trim trajectories can be carried out by gridding the set of possible values of $\bar{T}$; this set is bounded by the flight envelope constraint of the vehicle. While we are still unable to provide formal criteria for the selection of the "optimal" set of trim trajectories, results of recent work in quantized linear systems theory [16], as well as intuitive considerations suggest that a logarithmic spacing can be a better choice. The main idea behind logarithmic scaling is that a coarse control action is sufficient when the system is far from the equilibrium point; on the other hand a finer control action is needed to provide stability and performance near the equilibrium (see [17]).

4.4 Maneuvers

While a hybrid controller based only on trim trajectories is conceivable (this is a possible interpretation of the control architecture proposed in [18]), this generally results in "slow" transitions, as the system is required to stay in some sense close to the trim surface. Moreover, the absence of any information on the transient behavior can lead to undesirable effects, such as limit cycles. For aggressive maneuvering it is deemed necessary to better characterize trajectories that move "far" from the trim surface. In this paper, a maneuver is defined as a (finite time) *transition* between two trim trajectories. While this can be seen as a reductive definition, it leads to significant simplifications in the design of the control architecture. Note that the transition can also be from and to the same trajectory (e.g. acrobatic maneuvers like loops and barrel rolls can be considered as transitions from and back to straight and level flight). The execution of the maneuver results in a total configuration change g_{man}. We are more interested in the evolution on the subgroup H, and from the properties of trim trajectories we have that $h_{man} = (g_0^{-1})_{end} \, g_{man} \, (g_0)_{start}$. The reference trajectories can be generated using several kinds of methods, among which we can mention flight tests with human pilots, off-line solutions to optimal control problems, or real-time trajectory generation. A problem in the off-line generation of trajectories is the large amount of storage memory re-

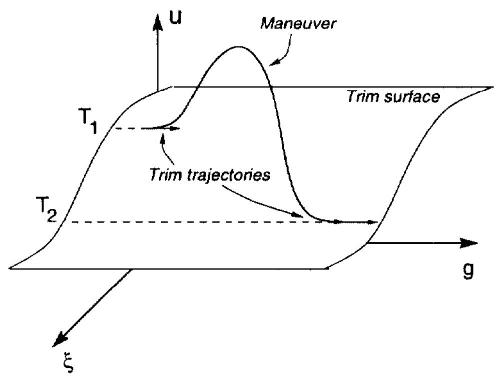

Figure 1: Trajectory primitives

quired; a possible solution is represented by some form of compression of the trajectory data. In this case, we have to identify some relevant parameters, on the basis of which the on-board processor can compute "easily", in real-time, a reference trajectory to track. A very efficient representation of trajectories can be achieved by exploiting the properties of the differentially flat approximation described in [10]. We recall that a system is differentially flat if we can find a set of outputs (the "flat" outputs) from the trajectory of which we can recover the full state and control trajectories. One possible choice of the flat outputs is composed by the position and heading angle: notice that these are the coordinates of the elements of the symmetry group H, and are hence a very appropriate choice for the present application. Moreover, it is easy to see that the parameter vector $\bar{T}$ can be derived from algebraic manipulation of the flat output and its first derivative.

5 Hybrid flight control system

We are now ready to discuss the details of the hybrid architecture. First we will introduce a general formalism, and then will specialize the formalism to our application.

5.1 General hybrid control system

For the formal definition of the controller structure, we will use (a simplified version of) the notation introduced in [19]. First of all we define a **controlled dynamical system** as the system: $\Sigma := [X, \Gamma, f, U]$ where the topological space X denotes the state space of Σ. The transition semi-group Γ is a topological semi-group with identity (this is an abstraction of time). Finally, f is the transition function, and is parameterized by the control set U_q. A **controlled general hybrid dynamical system** is defined as the system: $H_c := [Q, \Sigma, \mathbf{A}, \mathbf{G}, \mathbf{V}, \mathbf{C}, \mathbf{F}]$ where: Q is the (countable) set of discrete states; $\Sigma := \{\Sigma_q\}_{q \in Q}$ is the collection of constituent controlled dynamical systems. We will call the set $S := \bigcup_{q \in Q} \{q\} \times X_q$ the **hybrid state space** of H_c. $\mathbf{A} := \{A_q\}_{q \in Q}, A_q \subset X_q$ for each $q \in Q$, is the collection of autonomous jump sets; $\mathbf{G} := \{G_q\}_{q \in Q}$, where $G_q : A_q \times V_q \to S$ is the autonomous jump transition map, parameterized by the transition control set V_q, a subset of the collection $\mathbf{V} := \{V_q\}_{q \in Q}$; $\mathbf{C} := \{C_q\}_{q \in Q}, C_q \subset X_q$ for each $q \in Q$, is the collection of controlled jump sets; finally $\mathbf{F}_{q_{q \in Q}}$, where $F_q : C_q \to 2^S$, is the collection of controlled jump destination maps .

5.2 Specialization to the proposed architecture

In our flight control application, we will consider: $Q = Q_T \cup Q_M$ where the subscript T refers to trim trajectories, and M refers to maneuvers, as defined above; $X_q = \mathbf{R}^{n+1}$, where n is the dimension of the state space in the helicopter model, augmented by an additional "time" state τ (i.e. $\dot{\tau} = 1$); $\Gamma = \mathbf{R}_+$ (time) ; $f_q : X_q \times U \to TX_q$ is a Lipschitz function describing the continuous dynamics of each constituent subsystem, in the usual ODE form; $U \subseteq \mathbf{R}^m$, where m is the number of available (pseudo)-inputs; $A_q = \emptyset$ for $q \in Q_T$: no automatic jumps from trim trajectories; $A_q = \{x \in X_q | \tau \geq (\Delta t_{man})_q\}$ for $q \in Q_M$: automatic jumps at end of maneuvers; the corresponding jump map G_q takes the discrete states to a new $q \in Q_T$, and resets the time counter state τ to zero, while leaving the remaining components of x unchanged; $C_q = X_q$ for $q \in Q_T$: controlled jumps are always possible from trim trajectories; the corresponding jump map, parameterized by the control set $V_q \subseteq Q_M$, takes q into a new $q \in Q_M$, and resets the time counter τ to zero, again leaving the rest of x unchanged; $C_q = \emptyset$ for $q \in Q_M$: no controlled jumps from maneuvers.

5.3 Maneuver execution

Several researchers have worked extensively on the problem of trajectory tracking for nonlinear, possibly non-minimum phase systems with very interesting results (we can mention the recent papers [7, 8, 9, 10, 11]). Since the main focus of this paper is on trajectory planning and maneuver sequence generation, and the generated trajectories are by construction feasible trajectories for the nominal system, we will initially assume that perfect trajectory tracking is achieved by some lower level feedback controller, acting on the continuous control inputs. In any case, once we assign a feedback controller to each maneuver, the continuous evolution is then completely characterized.

5.4 Maneuver selection

At this point the design of the hybrid controller consists of the definition of a policy μ for selecting optimal jump times and destination (maneuver) from trim trajectories. We recall that all the relevant information while in a trim trajectory is defined by the hybrid automaton state and the "position" $h \in H$ at the current trajectory inception time ($\tau = 0$). On each trim trajectory $q \in Q_T$, the discrete control set can be identified with the subset $V_q \subseteq Q_M$ containing the indices of all the maneuvers that start at q. Moreover, the timing of the jump must be decided by the hybrid controller. The policy μ will then be a mapping $\mu : Q_T \times H \to Q_M \times \mathbf{R}_+$. Assume we want to control the system to the state $(\bar{q}, \bar{h})$, and define a running cost function $\gamma : Q \times H \to \mathbf{R}_+$, with $\gamma(\bar{q}, \bar{h}) = 0$. Given a policy μ we can define a total cost function:

$$J_\mu(q_0, h_0) := \int_{t_0}^{\infty} \gamma(q(t), h(t))\, dt \qquad (9)$$

where $h(t) = P_H[g(t)]$, P_H indicates a projection operator that maps elements of $SE(3)$ onto H, preserving position and heading angle, and the evolution of $g(t)$ is governed by the systems dynamics $\dot{g}(t) = g(t)\hat{\xi}(q,t)$. A policy μ is said to be *proper* if the above integral is finite for all initial conditions. Also, in the above we assumed that both autonomous and controlled jumps occur instantaneously, and have no cost penalty. The assumption that maneuvers are strictly finite time transitions between trim points (i.e. $\inf\{(\Delta t_{man})_q\}_{q \in Q_M} > 0$) ensures that there are finite switches in finite time, and that the resulting system trajectories are well defined. We assume that a proper policy exists, that is the system is controllable. It can be shown that this condition is satisfied if the set of design trim trajectories is rich enough to independently control the components of h (e.g. hover, straight and level flight, turn, climb, dive), and the maneuver set is such that it is possible to switch between any two trim trajectories in finite time (we can check this latter condition by solving for an appropriate shortest path problem). We are interested in computing the optimal policy μ^*, that is the policy that minimizes the total cost for all initial conditions. The ensuing optimal cost will be indicated by J^*. Following dynamic programming theory [20], it can be shown that the optimal cost satisfies the Bellman's equation:

$$J^*(q,h) = \min_{(\tau',q')} \left[\Gamma_T(q,h,\tau') + \Gamma_M(q',h') + J^*(q'',h'')\right] \tag{10}$$

where τ' is the delay before the commanded transition to $q' \in Q_M$, h' represents the position and heading at the start of the maneuver, and q'' and h'' represent the new state at the inception of the new trim trajectory. In the above equation, Γ_T and Γ_M indicate the cost associated with the trim and maneuver portions of the commanded transition. Moreover, the optimal control $(\tau',q')^*$ is the minimizer of Eq.(10). Nominal stability of the control algorithm is a consequence of the optimality condition. We notice that the optimization requires the solution of a mixed-integer program, with one continuous variable (τ'), and one discrete variable (q'). In general, the optimal cost function is not known. However, if an initial proper policy can be devised, approximate dynamic programming algorithms, such as value or policy iteration, can be used to improve on the initial policy, and possibly get to the optimal control strategy. Moreover, since the dimension of the state space has been reduced to one discrete variable and four continuous ones, neuro-dynamic programming approximation techniques for a compact representation of the cost function can be effectively used, making the control algorithms suitable for real-time applications [21].

5.5 Application example

As an application example, we consider the minimum time optimal control problem, in an obstacle-free environment. We want to take the helicopter to hover in a neighborhood of the origin in minimum time, under the constraint of the allowable maneuvers. In this case the running cost function is:

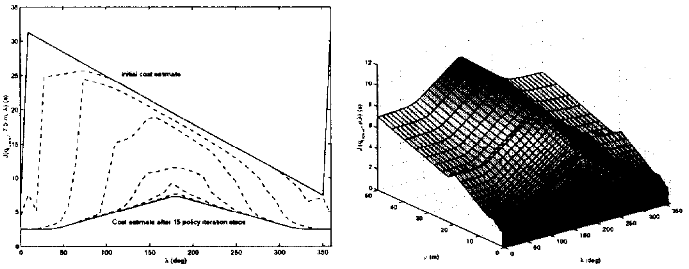

Figure 2: Value iteration results and optimal cost

$$\gamma(q,h) = \begin{cases} 0 & \text{for } (q,h) = (q_{hover}, [\bar{x}, \cdot]^T), ||\bar{x}|| < \varepsilon \\ 1 & \text{otherwise} \end{cases} \tag{11}$$

The radius of the target zone ε can be made arbitrarily small, but must be strictly positive, at least in the current implementation of this architecture, because of truncation (finite number of trajectory primitives), and computational issues (continuity at the optimum). As simplifying assumptions, we will consider an obstacle-free environment, and will consider only trajectories in the horizontal plane. In this case the problem has an axial symmetry, and the relevant information in the outer state vector can be reduced to the scalar quantities ρ and λ, that is the distance and the line-of-sight angle to the target. In the example, the design trim trajectories collection is defined by:

$$\begin{aligned}(V,\dot{\psi},\gamma,\beta) \in\ & \{0, 1.25, 2.5, 5, 10\,\text{m/s}\} \times \\ & \times\{-1, 0.5, 0, 0.5, 1\,\text{rad/s}\} \times \\ & \times\{0\,\text{rad}\} \times \{0\,\text{rad}\}\end{aligned}$$

Reference maneuvers are computed for transition between all trim trajectories. An initial proper control policy, based on heuristics, can easily be derived (i.e. stop the helicopter, turn facing the target, move slowly towards the target). Application of a value iteration algorithm provides convergence in the evaluation of the optimal cost-to-go to within one hundredth of a second in 15 iterations. In this simple application example, the evaluation of the optimal cost can be carried out off-line (see fig. 2). This is a consequence of the fact that the environment is unchanging and perfectly known. On the other hand, the evaluation of the optimal control, that is minimization of J defined in eq. (10), has to be done in real-time. The computation of the optimal control in this example requires only a few hundredths of a second on a Pentium-class CPU, and is therefore implementable on current on-board computer systems for small aerial vehicles. Examples of trajectories obtained by simulation are shown in fig. 3. In these figures, the height of the stems represents the velocity of the vehicle; moreover, solid lines and circle symbols indicate transitions.

6 Conclusions and future work

In this paper we have presented an outline of a new architectural concept for aggressive maneuvering of autonomous

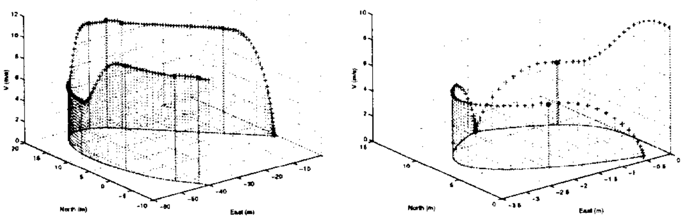

Figure 3: Simulated trajectory and velocity profile, starting from a high speed turn away from the target (left), and from high speed flight over the target

vehicles. The architecture seems promising, however several issues must be explored. In the first place we have to mention the stability and performance robustness of the hybrid control algorithm in the face of uncertainty in the environment and in the model. In this paper we have assumed perfect tracking of the nominal trajectory: this will not be achieved in reality, and the stability of the trajectory generated by the hybrid automaton has to be analyzed. The selection of the trajectory primitives is currently done manually: it would be desirable to obtain formal criteria defining the "optimal" choice of primitives, trading off the complexity of the resulting automaton with the achievable performance. A dynamic resizing of the automaton is also conceivable: in critical situations, when a decision has to be taken in a very short time, the automaton could be reduced to a few maneuvers, whereas in a more secure situation the set of possible maneuvers could be expanded. Extensions of the control algorithm could be made to allow multi-vehicle operations, as well as flight in an unknown environment (i.e. obstacle avoidance). Moreover, different kinds of objective functions can be considered, such as threat evasion, or target acquisition and payload delivery. Finally, a similar structure can be thought for the mission planning level, for executing functions as objective prioritization, scheduling and allocation to multiple agents. All these questions are the object of current research efforts.

References

[1] J. T. Betts. Survey of numerical methods for trajectory optimization. *AIAA Journal of Guidance, Control, and Dynamics*, 21(2):193–207, March-April 1998.

[2] R. F. Stengel. Toward intelligent flight control. *IEEE Transactions on Systems, Man and Cybernetics*, 23(6):1699–1717, November-December 1993.

[3] J. W. Jackson and P. E. Crouch. Dynamic interpolation and application to flight control. *AIAA Journal of Guidance, Control, and Dynamics*, 14(4):814–822, July-August 1991.

[4] P.K.A. Menon, E. Kim, and V.H.L. Cheng. Optimal trajectory synthesis for terrain-following flight. *AIAA Journal of Guidance, Control, and Dynamics*, 14(4):807–813, July-August 1991.

[5] T. J. Koo, F. Hoffmann, B.Sinopoli, and S. Sastry. Hybrid control of an autonomous helicopter. In *IFAC Workshop on Motion Control*, 1998.

[6] C.P. Sanders, P.A. DeBitetto, E. Feron, H.F. Vuong, and N. Leveson. Hierarchical control of small autonomous helicopters. In *37th IEEE Conference on Decision and Control*, 1998.

[7] C. Phillips, C.L. Karr, and G. Walker. Helicopter flight control with fuzzy logic and genetic algorithms. *Engineering Applications of Artificial Intelligence*, 9(2):175–184, 1996.

[8] J. Hauser and R. Hindman. Aggressive flight maneuvers. In *IEEE Conference on Decision and Control*, 1997.

[9] D.H. Shim, T. J. Koo, F. Hoffmann, and S. Sastry. A comprehensive study of control design for an autonomous helicopter. In *37th IEEE Conference on Decision and Control*, 1998.

[10] T.J. Koo and S. Sastry. Output tracking control design of a helicopter model based on approximate linearization. In *IEEE Conference on Decision and Control*, 1998.

[11] S.A. Al-Hiddabi and N.H. McClamrock. Output tracking for nonlinear non-minimum phase VTOL aircraft. In *IEEE Conference on Decision and Control*, 1998.

[12] W. Johnson. *Helicopter Theory*. Princeton University Press, 1980.

[13] M. J. Van Nieuwstadt and R. M. Murray. Real-time trajectory generation for differentially flat systems. *International Journal of Robust and Nonlinear Control*, 8(11):995–1020, September 1998.

[14] S.K. Kim and D.M. Tilbury. Mathematical modeling and experimental identification of a model helicopter. In *AIAA-98-4357*, 1998.

[15] R.M. Murray, Z.Li, , and S.S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[16] N. Elia and S. Mitter. Quantized linear systems. Submitted to the 1999 IEEE Conference on Decision and Control.

[17] R.W. Brockett. Minimum attention control. In *36th IEEE Conference on Decision and Control*, 1997.

[18] M.W. McConley, B.D. Appleby, M.A. Dahleh, , and E.Feron. A computationally efficient Lyapunov-based scheduling procedure for control of nonlinear systems with stability guarantees. *IEEE Transactions on Automatic Control*, December 1999.

[19] M. S. Branicky. *Studies in Hybrid Systems: Modeling, Analysis, and Control*. PhD thesis, Massachusetts Institute of Technology, 1995.

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

[21] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.