LIDS-P-2468

Robust Hybrid Control for Autonomous Vehicle Motion Planning

Emilio Frazzoli * Munther A. Dahleh † Eric Feron ‡

December 5, 1999

Abstract

The operation of an autonomous vehicle in an unknown, dynamic environment is a very complex problem, especially when the vehicle is required to use its full maneuvering capabilities, and to react in real time to changes in the operational environment. A possible approach to reduce the computational complexity of the motion planning problem for a nonlinear, high dimensional system, is based on a quantization of the system dynamics, leading to a control architecture based on a hybrid automaton, the states of which represent feasible trajectory primitives for the vehicle. This report focuses on the feasibility of this approach: the structure of a Robust Hybrid Automaton is defined and its properties are analyzed. Algorithms are presented for time-optimal motion planning in a free workspace, or in the presence of fixed or moving obstacles. A case study involving a small autonomous helicopter is presented: a nonlinear control law for maneuver execution is provided, and a robust hybrid automaton is constructed. Simulations showing the effectiveness of the approach are presented and discussed.

1 Introduction

In the past few years considerable interest has been shown and relevant resources have been devoted, from industry, government and academia, to the design, development and operation of autonomous aerial, underwater, and ground vehicles. The possibility of removing human pilots from danger, and the size and cost advantages of autonomous vehicles are indeed very attractive, but very often have to be compared with the performance that can be attained by human-piloted vehicles, in terms of mission capabilities, efficiency, and flexibility.

*Research Assistant, Laboratory for Information and Decision Systems, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, email:frazzoli@mit.edu

†Professor, Laboratory for Information and Decision Systems, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, email: dahleh@lids.mit.edu

‡Associate Professor, Laboratory for Information and Decision Systems, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, email: feron@mit.edu

The operation of an autonomous vehicle in an unknown, dynamic and potentially hostile environment is a very complex problem, especially when the autonomous vehicle is required to use its full maneuvering capabilities, and to react in real time to changes in the operational environment. A common way of dealing with highly complex systems is via a hierarchical decomposition of the activities to be performed by the autonomous vehicles and consequently the introduction of a hierarchy of control and decision layers [1, 2].

In the case of an autonomous vehicle the plant dynamics are inherently continuous, and as such are described by ordinary differential equations. On the other hand, digital computers are used for control purpose, which means that the overall control system will be discrete in nature. The control layers that are closest to the plant are usually characterized by high bandwidth, and in some cases are seen, for design purposes, as sharing the continuous nature of the of the underlying plant. Higher control layers are most often designed as discrete, logical decision-making agents. Systems that include both discrete and continuous dynamics are usually referred to in the literature as hybrid systems. Hybrid control systems have been the object of a very intense and productive research effort in the recent years, which has resulted in the definition of very general frameworks [3, 4]. General hybrid systems, derived from arbitrary hierarchical decompositions, however, can be extremely hard to analyze and verify, and only limited results can be obtained [5, 6].

On the other hand, it could be convenient to design the hybrid system in such a way that it offers safety and performance guarantees by construction, at least in an idealized situation. This reflects the designer's insight into the nominal behavior of the system [7, 8, 9]. Consequently, the analysis and verification problems of the hybrid system are translated to a robustness analysis problem, which can be solved using the relevant tools from systems and control theory .

In this report, our main objective is the definition of a robust control architecture, and algorithms, to address the motion planning problem for an autonomous vehicle, in the presence of obstacles, and exploiting to the maximum extent the vehicle's dynamics. Even though our main focus is control of autonomous vehicles, the concepts that we will introduce can be used profitably for control of a large class of nonlinear systems: the presentation of the control architecture will be kept at a general level. In section (2) the main component of our controller, that is a robust hybrid automaton, is introduced. In the following section (3), control policies on the hybrid automaton are developed, both for an empty workspace and in the presence of obstacles. A case study, in section (4) concentrating on motion planning for a small autonomous helicopter will demonstrate the general concepts on this particularly challenging application. Trajectory tracking control, optimal control in a free environment, and motion planning in the presence of moving obstacles will be addressed and simulation results will be provided.

2 Robust Hybrid Automaton

A possible approach to reduce the computational complexity of the motion planning problem for a nonlinear, high dimensional system, is based on a discretization of the system dynamics, in the sense that we restrict the feasible nominal system trajectories to the family of time-parametrized curves that can be obtained 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. This report will focus on studying the practical feasibility of this approach.

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 stable hierarchical system, and for developing the tools to analyze robustness in the presence of uncertainty in the process as well as in the environment.

We want to characterize trajectory primitives in order to: (i) capture the relevant characteristics of the vehicle dynamics; (ii) allow for the creation of complex behaviors from the interconnection of primitives (we want to obtain "good" approximations to optimal solutions) (iii) determine the *minimal* set of key parameters identifying the state of the system: this is even more important for extension to multi-vehicle operations, or more complex systems.

In the following we will define the class of systems we want to control, and, accordingly, we will give a characterization of the trajectory primitives we will consider. This will be used to present the Robust Hybrid Automaton structure.

2.1 System dynamics

We will consider a nonlinear system the dynamics of which are described by the ODE:

$$\frac{dx}{dt} = f(x, u, w) \tag{1}$$

where $x \in X$ is the state, belonging to a smooth manifold X, locally diffeomorphic to $\mathbb{R}^n$, and u and w represent respectively the control and disturbance input signals, taking values in the sets $U \subseteq \mathbb{R}^m$, $W \subseteq \mathbb{R}^l$; both signals are assumed to be bounded with respect to some norm p, i.e. $u(\cdot) \in \mathcal{U} \subseteq \mathcal{L}_p^m$, $w(\cdot) \in \mathcal{W} \subseteq \mathcal{L}_p^l$. Finally, the function $f : X \times U \times W \to TX$ is assumed to be locally Lipschitz in its arguments.

Let us concentrate first on the nominal system:

$$\dot{x} = f_0(x, u) = f(x, u, 0) \tag{2}$$

that is the system obtained from (1) when the disturbance input is identically zero.

We can define equilibrium points for the nominal system (2) as the points $(\bar{x}, \bar{u})$ for which $f_0(\bar{x}, \bar{u}) = 0$. In general, we will be more interested in systems that have symmetries, for which we can define the notion of relative equilibrium.

2.2 Symmetries and relative equilibria

Roughly speaking, a symmetry on the system (2) is a group action on the state that leaves the dynamics unchanged (see [10] for a precise definition for mechanical systems). A simpler definition that is enough for the purpose of this report is given in the following.

Assume that the state space has a group structure $(X, \cdot)$, and consider the (smooth) automorphism $\psi_y : X \to X$, parametrized by an element y of the subgroup $Y \subseteq X$, such that the state $x \in X$ is transformed into $x \cdot y$. Given the initial conditions (x_0, t_0), let the time-parametrized curve $\bar{\phi}(\cdot, x_0, t_0)$ be the resulting trajectory of the nominal system (2) when the control input is kept at a given constant value, i.e. $u(t) = \bar{u}$. We can define as the push-forward of $\bar{\phi}$ the curve $(\psi_y \circ \bar{\phi}) : \mathbb{R} \to X$ obtained by applying ψ_y to $\bar{\phi}$. If for all initial conditions and for all $y \in Y$ the following holds:

$$\psi_y \circ \bar{\phi}(t, x_0, t_0) = \bar{\phi}(t, x_0 \cdot y, t_0) \tag{3}$$

then Y is a symmetry group for the system (2). A trajectory $\bar{\phi}$ that evolves along a symmetry group is known as a *relative equilibrium* of the system. For controlled systems, since the control input is constant, relative equilibria are also known as *trim trajectories*; the collection of all possible trim trajectories defines a manifold $\mathcal{S} \subset X$, denoted as the trim surface. To each relative equilibrium corresponds a control input $\bar{u}$ and an integral of motion, generally known as momentum (Noether's theorem [10]). It is also clear that relative equilibria include trivially all the equilibrium points of the system.

A very simple example of a system with symmetries is a system with integrators:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f_2(x_2, u) \end{aligned} \tag{4}$$

where $x_1, x_2 \in \mathbb{R}^n$, and the group operation we are interested is the usual vector addition. It is evident that a translation $\psi_{\Delta x} : (x_1, x_2) \mapsto (x_1 + \Delta x_1, x_2)$ does not change the dynamics of the system. Relative equilibria are all trajectories for which $\dot{x}_2 = f_2(\bar{x}_2, \bar{u}) = 0$. The momentum can be identified in the constant vector $\bar{x}_2$, and the dynamics of x_1 on a trim trajectory is given by $x_1 = x_1(t_0) + \bar{x}_2(t - t_0)$.

A more interesting kind of symmetry, that is invariance to translation and rotation about a vertical axis, is exhibited by a large class of mechanical systems. This is true of most human built vehicles: there is a simple reason behind this, that is vehicles are built such that once we learn how to operate them at some location (e.g. at the driving school grounds), we can apply the same skills to drive anywhere in the world.

In the following we will focus on the definition of a control architecture for autonomous vehicles, however the concepts and methods are valid and can be used for systems with multiple equilibria, and possibly with relative equilibria.

2.3 Autonomous vehicle dynamics

The dynamics of a large class of small autonomous vehicles can be adequately described by the rigid body equations [11]. The configuration of the vehicle 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:

$$g = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \tag{5}$$

where $R \in SO(3)$ is a rotation matrix and $p \in \mathbb{R}^3$ is a translation vector. The kinematics of the rigid body are determined by

$$\dot{g} = g\hat{\xi} \tag{6}$$

where $\hat{\xi}$, denoted as *twist*, is an element of the Lie algebra $\mathfrak{se}(3)$ associated with $SE(3)$. A matrix representation of an element $\hat{\xi} \in \mathfrak{se}(3)$ is

$$\hat{\xi} = \begin{bmatrix} \hat{\omega} & v \\ 0 & 0 \end{bmatrix} \tag{7}$$

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 $u \in \mathbb{R}^3$ The full state of the vehicle as a rigid body will then be represented by $x = (g, \hat{\xi})$, with $X = SE(3) \times \mathfrak{se}(3)$. The dynamics equations, in matrix notation, will be given by:

$$J_b\dot{\omega} = -\omega \times J_b\omega + M_b(g, \xi, u, w) \tag{8}$$

$$m\dot{v} = -\omega \times mv + F_b(g, \xi, u, w) \tag{9}$$

where J_b and m are the vehicle's inertia tensor and mass, and M_b and F_b represent the torques and forces in body axes, which are in general a function of the vehicle state $x = (g, \hat{\xi})$, of the control inputs u, and of the disturbances w. Note that in the above we have no assumption on the characteristics of the forces acting on the vehicle (i.e. we do not require potential forces).

The dynamics of a vehicle, including cars, aircraft, ships, etc., under fairly reasonable assumptions, (such as homogeneous and isotropic atmosphere, and constant gravity acceleration for an aircraft) are invariant to translation and rotation about a vertical axis, i.e. an axis parallel to the local gravitational acceleration. If this is the case, the subgroup $H \subset SE(3)$, composed of translation and rotations about the vertical axis is a symmetry group for the vehicle dynamics. As a consequence of the symmetry property, if $h \in H$ then it follows that :

$$\dot{\xi}(g, \xi, u) = \dot{\xi}(gh, \xi, u) \tag{10}$$

An element $h \in H$ is completely described by the translation vector $p \in \mathbb{R}^3$ and the heading angle $\psi \in [0; 2\pi)$.

2.4 Equilibrium points and trim trajectories

The simplest possible motion primitive is trivially represented by equilibrium points. In a system with multiple equilibrium points each equilibrium point can be chosen as a trajectory primitive. A closely related and more interesting class of primitives is given by trim trajectories. In an autonomous vehicle setting, these can be seen 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 $g = \bar{g}\exp(\eta t)$ of a constant rotation $\bar{g}$ and a screw motion $h(t) \in H$, given by the exponential of an element η of the Lie sub-algebra $\mathfrak{h} \subset \mathfrak{se}(3)$. This screw motion corresponds in the physical space to a helix traversed at a constant sideslip angle. For aerial vehicles, such helices are usually described by the parameter vector $\bar{T} := [V, \gamma, \dot{\psi}, \beta]$, where V is the magnitude of the velocity vector, γ is the flight path angle, $\dot{\psi}$ is the turning rate and finally β is the sideslip angle [12].

Clearly, if we restrict the motion of the vehicle to a horizontal plane (as in the case of a ground vehicle, a surface vessel, or an aircraft at constant altitude), the class of trim trajectories becomes simply the set of circle arcs on the plane, including infinite radius turns, that is straight lines. Note also that while for many kinds of vehicles, such as fixed wing aircraft, β is usually assumed to be zero, or very small (no skidding, or coordinated flight), this is not necessarily true for vehicles like helicopters, especially for low velocity regimes (e.g. sidesteps and backwards motion can be accomplished by helicopters).

It should be mentioned that usually 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. 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 operating envelope constraints of the vehicle.

This class of trajectory primitives has been used widely to construct switching control systems, in which point stabilization is achieved by switching through a sequence of controllers progressively taking the system closer to the desired equilibrium [13, 14, 15]. The ideas of gain scheduling and of LPV system control can also be brought into this class [16], as well as some innovative integrated guidance and control systems successfully used in UAV applications [17].

However, such a design choice generally results in relatively poor performance, and 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.

2.5 Maneuvers

For more aggressive maneuvering it is deemed necessary to better characterize trajectories that move "far" from the trim surface. In this report, a maneuver is defined as a (finite time) *transition* between two trim trajectories, for the nominal system (2).

While this can be seen as a reductive definition of what is considered a maneuver in the common language, 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. in the case of aircraft acrobatic maneuvers like loops and barrel rolls can be considered as transitions from and back to straight and level flight, and in the case of cars a lane change is a transition from and back to forward motion)

The execution of the maneuver results in a total configuration change g_m, that is $g(t_m) = g_m g(0)$ if t_m indicates the maneuver duration. We are more interested in the evolution on the subgroup H,

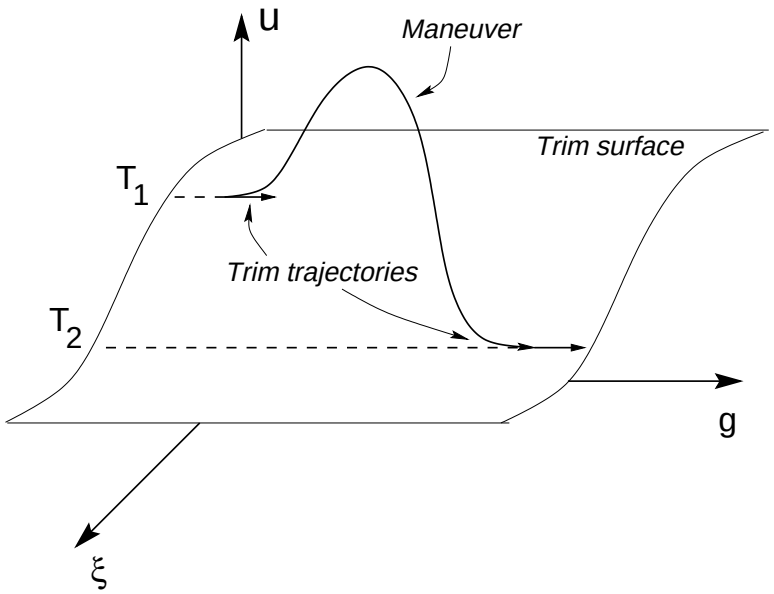

Figure 1: Trajectory primitives

and from the properties of trim trajectories we have that $h_m = \bar{g}_{end}^{-1}\, g_m\, \bar{g}_{start}$. We will not discuss the details of how to generate the reference trajectories: several methods can be used depending on the application at hand, the desired performance, and the available computing, simulation and experimental resources. Among thess methods we can mention actual tests or simulations 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 required; 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 differentially flat systems [18, 19].

We notice that the design of the reference trajectories, along with the tracking control introduced in the next section, has to be carried out in such a way to ensure that the vehicle does not violate constraints on its dynamic envelope (e.g. maximum velocity) [20]. Thus in this sense the objective of envelope protection is ensured implicitly by the maneuver definition.

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 on the symmetry group. Without loss of generality, we can define all trajectory primitives as starting at the identity element e_H of the symmetry group H.

2.6 Tracking control and disturbance rejection

In the preceding sections we defined the trajectory primitives as feasible, possibly optimal (for some cost) trajectories for the *nominal* system, that is when the disturbance signal w is identically zero. In real applications, we will not be able to achieve exactly the reference trajectories defined by the primitive library, because of deviations in the initial conditions, noise in the measurements,

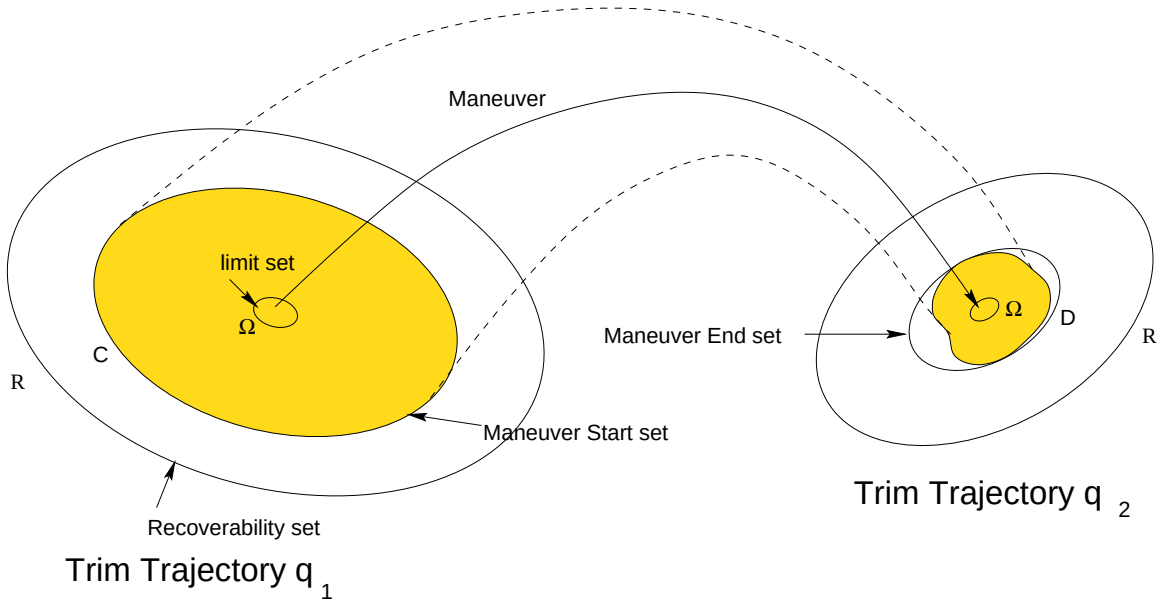

Figure 2: Invariant set definitions

unmodeled dynamics and modeling errors, and exogenous inputs. We will therefore need to examine the behavior of the system at non-nominal conditions, and make sure that the resulting system trajectories are in some sense "close" to the trajectories of the nominal system. In general, this requires some form of feedback control, complementing the feed-forward open loop reference input trajectory stored along with the reference state trajectory.

The reference trajectory will be completely determined in terms of nominal state and control histories by the primitive being executed, its inception time, and initial H configuration. A feedback control policy will then be a function $\upsilon : \mathbb{R} \times X \times \mathbb{R} \times H \rightarrow U$, designed to track (or regulate to) the nominal trajectory. Once a feedback control law is associated with the system (1) it is transformed into a system of the form:

$$\dot{x} = f(t, x, \upsilon(t, x, t_0, h_0), w) \tag{11}$$

in which the only input that is free to vary is the disturbance input w.

The robustness characteristics of equilibrium points and of trim trajectories can be expressed in terms of invariant sets. A set $M \subseteq X$ is said to be a (right)-invariant set if whenever $x_0 \in M$, and for all disturbance signals $w \in \mathcal{W}$, we have, for all $t > t_0$:

$$\psi_{\exp(-\eta(t-t_0))} \circ \phi_{\upsilon,w}(t, x_0, t_0) \in M \tag{12}$$

where $\phi_{\upsilon,w}(\cdot, x_0, t_0)$ describes the trajectory of the system under the action of the control policy υ and disturbance w, and with initial conditions $x(t_0) = x_0$. Invariant sets are, properly speaking, cylinders with axis $h\eta$. However, in the following we will refer to the section at $h = e_H$. We will call a set $\bar{\Omega}_q$ the limit set of a trim trajectory q the smallest invariant set associated with that trajectory. Accordingly, we will define the recoverability set $\bar{\mathcal{R}}_q$ of a trim trajectory as the largest set for which there exists a finite time $\tilde{t}$ such that for all initial conditions $x_0 \in \bar{\mathcal{R}}_1$, and for all disturbance signals $w \in \mathcal{W}$, the system enters the limit set, that is if :

$$\psi_{\exp(-\eta(t-t_0))} \circ \phi_{\upsilon,w}(t, x_0, t_0) \in \Omega_q, \ \forall\, t > t_0 + \tilde{t} \tag{13}$$

In general, the exact determination of the sets $\bar{\mathcal{R}}_q$ and $\bar{\Omega}_q$ presents a very difficult challenge. However, often it is possible to compute conservative estimates, in the sense that we can compute a set $\mathcal{R}_q \subseteq \bar{\mathcal{R}}_q$ such that for all initial conditions in $\mathcal{R}_q$ the system trajectory will enter a set $\Omega_q \supseteq \bar{\Omega}_q$ after a finite time, and stay in Ω_q thereafter.

It is obviously of interest to design a control law in such a way to have a large $\mathcal{R}_q$, and a small Ω_q. In the case in which the control law provides global stability, $\mathcal{R}_q$ will coincide with X, and in the case in which we have asymptotic stability, Ω_q will collapse to the trajectory itself.

Similar concepts, close in nature to Lyapunov stability theory, cannot be defined for the maneuvers, since these are by definition objects with a finite time horizon. Instead we will use a concept more closely related to Poincarè maps. We will call a set $D \in X$ the image of a set C under the maneuver q if for all $x_0 \in C$, and for all disturbance signals $w \in \mathcal{W}$ (supported on $[t_0, t_0 + \Delta t_{m_q}]$),

$$\phi_{\upsilon,w}(t_0 + \Delta t_{m_q}, x_0, t_0) \in D \tag{14}$$

The objective of the control law in this case is to make the ending set D as small as possible for a given starting set C. Notice that we are not directly interested in the transient behavior of the system during the execution of the maneuver, as long as we can ensure that at the end of the maneuver the state enters the set D.

2.7 Robust Hybrid control system

We are now ready to discuss the details of the control architecture. It should be clear by now that the controlled system will include both continuous and discrete dynamics, thus belonging to the realm of hybrid control. In the following we will present the definition of a hybrid system that is based on the general model in [3].

The hybrid control system we are concerned with is described by the following objects:

- $Q := Q_M \cup Q_T$ is the discrete set of the index state. The values of Q identify the trajectory primitive being executed; as such, Q is assumed to be a finite set. The subscript T and M indicate, respectively, trim trajectories and maneuvers;
- H is the symmetry group, identifying the position of the current trajectory primitive;
- $T = \mathbb{R}$: we augment the reference state by a clock, or timer state;
- X is the state space of the continuous system;
- f is the Lipschitz function describing the continuous system dynamics in the usual ODE form;
- $\eta := \{\eta_q, q \in Q_T\}$, where each η_q describes the motion on the trim trajectory q;
- $\upsilon := \{\upsilon_q, q \in Q\}$ is the set of control laws designed for each trajectory primitive; we assume that each υ_q is a Lipschitz function;
- $\mathbf{C} := \{C_q, q \in Q_M\}$: the collection of sets from which we can initiate maneuvers (controlled jump set);

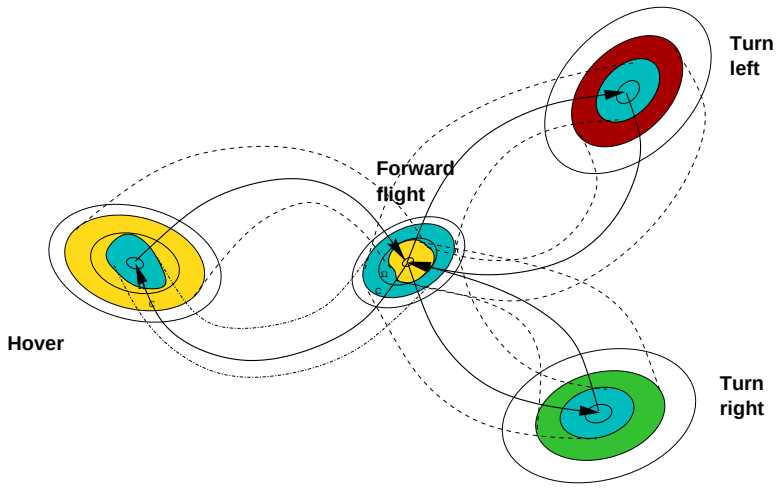

Figure 3: Robust Hybrid Automaton (simplified)

- $\mathbf{D} := \{D_q, q \in Q_M\}$: the collection of sets at which maneuvers are terminated;
- $R := \{\mathcal{R}_q, q \in Q_T\}$: the collection of recoverability sets for each trim trajectory;
- $\Omega := \{\Omega_q, q \in Q_T\}$: the collection of limit sets for each trim trajectory;
- $\mathbf{A} = \{A_q = (q_{new}, \Delta t, \Delta h), q\ in Q_M\}$: autonomous jumps occur during maneuver execution, when the timer state $\tau \in T$ reaches the value ΔT. The state is reset such that $q \leftarrow q_{new}$, $h \leftarrow \Delta h \cdot h$, $\tau \leftarrow 0$;
- $V := \mathbb{R} \times Q_M$: is the hybrid control set, determining the controlled jump execution. Controlled jumps can only be executed from trim trajectories; given a hybrid control $(\Delta t_{coast}, q_{new})$, the jump occurs when $\tau \in T$ reaches Δt_{coast}, and the state is reset such that: $q \leftarrow q_{new}$, $h \leftarrow \exp(\eta_q \Delta t_{coast}) \cdot h$, $\tau \leftarrow 0$;

We can graphically depict the hybrid automaton as a directed graph, where the nodes represent the trim trajectories, and the edges represent the maneuvers. Each edge can be labeled with a cost corresponding to the maneuver duration.

The introduction of a time clock state could seem pointless, as we will in the following assume $\dot{\tau} = 1$ (when not switching). This corresponds to a classical formulation of a trajectory tracking control. The reason for introducing the clock state is that we want to allow for *maneuver regulation* [21, 17]. In that case, we will have to introduce some condition of the form $\dot{\tau}_i = \theta_i(\tau_i, x, x_{ref,i}) \geq \epsilon > 0$, to ensure that the system does not move "backwards" along a trajectory.

2.8 Well-posedness, consistency, and controllability

When dealing with systems of the form of (1), where the right-hand side is not continuous, care must be taken to ensure that the system is well-posed, that is that a unique solution exists. In our case well-posedness is ensured by the fact that the system is piecewise continuous, and the maneuvers are a finite set of primitives with a finite time duration. As a consequence in every finite time interval there will be a finite number of switches, or discontinuities in the feedback map.

We say that the automaton is *consistent* if for all $q \in Q_T$ the following conditions hold:

1. $\Omega_q \subseteq C_l,\ \forall l \in L_q$;
2. $\bigcup_{p \in P_q} D_p \subseteq \mathcal{R}_q$;
3. $\bigcup_{p \in P_q} D_p \subseteq C_l,\ \forall l \in L_q$,

where $L_q, P_q \subset Q_M$ are respectively the set of indices of maneuvers leaving and arriving at the trim trajectory q. Notice that if the first and second conditions are satisfied, the third one can be satisfied by adequately extending the time duration of the maneuvers which originally violated it.

If an automaton is consistent then any sequence of hybrid controls will generate a trajectory which remains "close" to the nominal trajectory, in the sense of the invariant sets defined in the previous sections.

The full state of the system will then be described by (q, h, τ, x), where $q \in Q, h \in H, \tau \in \mathbb{R}, x \in X$. We will say that the system is in the hybrid state (q, h, τ) if $x \cdot h^{-1} \in \bigcap_{l \in L_q} C_l \subseteq R_q$, that is if the continuous state is inside the starting sets for all the maneuvers leaving the current trim trajectory.

Once we have established that the hybrid system is well posed and consistent, we have to ensure that it is possible to steer the system from any initial condition (q, h) to any desired location $\bar{h} \in H$, and at any desired operating condition $\bar{q} \in Q_T$. If this is possible in finite time then the hybrid system is completely controllable.

It is clear that a necessary condition for controllability is that the directed graph describing the automaton be fully connected. Moreover, the set of trim trajectories must be rich enough that by interconnecting an appropriate sequence of them we cover the group H.

In the case of a system with integrators, this translates simply to the requirements that the set $\{\eta_q\}, q \in Q_T$ is a complete basis for $\mathbb{R}^n$.

For an autonomous vehicle, it can be verified that the minimum collection of trim trajectories is composed by just two elements, described by the parameters $\left\{V_1, \dot{\psi}_1, \gamma_1, \beta_1\right\}$, $\left\{V_2, \dot{\psi}_2, \gamma_2, \beta_2\right\}$, for which: $V_1, V_2 > 0$, $\dot{\psi}_1 \neq \dot{\psi}_2$, and $\gamma_1 < 0 < \gamma_2$. It is clear that this would be just a minimum set of trim trajectories to ensure controllability: for practical applications, the set of trim trajectories will be much richer. Notice that the "uncontrollable" car in [22], which can only turn left with different turning radii, is in fact controllable according to our definition.

3 Motion planning

The hybrid control architecture lends itself to computationally efficient solutions of many problems of interest for practical applications; the same problems cannot be dealt with using standard nonlinear control theory. Most often the best results that are achievable using nonlinear control theory involve stability or tracking performance. Ensuring the feasibility of solutions with respect

to non-convex constraints such as obstacles in the physical workspace is in most cases impossible in this framework alone. At the same time when we are dealing with systems whose dynamics is important, we cannot just deal with a simplified kinematic model, or with a discretized, logic-based controller, as the resulting trajectory will in general be unfeasible for the system.

The price that we have to pay in using the hybrid automaton is the sub-optimality of the computed solutions, owing to the fact that the stored trajectory primitives do not represent the whole dynamics of the system. However, the number of trajectory primitives stored in the automaton can be increased, depending on the available memory and computational resources, so the sub-optimality gap can be reduced to a point were it is not noticeable for practical purposes. Moreover, very often a sub-optimal solution which is computable on-line can be worth more that an optimal solution that requires computational resources only available as off-line planning.

3.1 Optimal control in a free environment

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 $\mu : Q_T \times H \to \mathbb{R}_+ \times Q_M$. Assume we want to control the system to the state $(\bar{q}, \bar{h})$, and define a running cost function $\gamma : Q \times H \times \mathbb{R}_+ \to \mathbb{R}_+$, with $\gamma(\bar{q}, \bar{h}, \cdot) = 0$. Given a policy μ we can define a total cost function:

$$J_\mu(q_0, h_0) := \int_{t_0}^{\infty} \gamma\left(q(t), h(t), \tau(t)\right) dt \tag{15}$$

where the system dynamics are governed by the continuous evolution and the jump rules given in the previous sections.

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. $\min\{(\Delta t_{man})_q\}_{q \in Q_M} > 0$) ensures that there are finite switches in finite time, and that the resulting system trajectories are well defined. If the system is controllable, then a proper policy exists.

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 [23], it can be shown that the optimal cost satisfies Bellman's equation:

$$J^*(q, h, \tau) = \min_{(\tau', q')} \left[, _T(q, h, \tau') + , _M(q', h') + J^*(q'', h'') \right] \tag{16}$$

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, $, _T$ and $, _M$ 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.(16). Nominal stability of the control algorithm is a consequence of the optimality condition; from the consistency conditions on the automaton we also have that the system will be driven to the limit set $\Omega_{\bar{q}}$.

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 [24].

3.2 Motion planning in the presence of obstacles

A very important problem, especially when controlling autonomous vehicles, is represented by motion planning in the presence of fixed or moving obstacles. We are using the expression "motion planning" as opposed to the traditional "path planning" because we want to emphasize the role of the dynamics of the system, or of non-holonomicity constraints, on the allowable feasible trajectories; this problem is also known in the literature as kinodynamic planning[25].

One of the fundamental conceptual steps in addressing a problem is the selection of the appropriate representation of the system state, constraints and of the decision variables. In the path planning literature, the most commonly encountered object underlying the algorithm design is the configuration space, that is the set of all possible collision-free configurations for the robot [26, 27]. In [25], the author suggests that a more appropriate representation for a large class of motion planning problem is via the state space, since this encodes the additional information related to the system dynamics. However, the state space of non-trivial systems is typically very large, and the "curse of dimensionality" makes the solution of motion planning problems in such large-dimension spaces computationally intractable.

The alternative that we propose, through the introduction of our hybrid control architecture, can be seen as a maneuver automaton space. Using this representation, the hybrid automaton already encodes all the relevant information about the system dynamics and dynamic constraints (such as non-holonomicity). The dynamic constraints must then be complemented by the configuration constraints. We can define a set $\mathcal{Q} \subseteq Q \times H$ of primitives and starting configurations, such that the resulting trajectory is collision free. The set $\mathcal{Q}$ is the direct counterpart of the traditional configuration space $\mathcal{C}$, but in addition completely encodes the system dynamic constraints.

To check whether a given trajectory is collision-free, we have to check that the appropriate invariant sets do not intersect with the obstacles. In the case of trim trajectories, the relevant set is the controlled jump set $\bigcup_{l \in L_q} C_l$. In the case of maneuvers, this would be the set spanned by the image of C_q over the maneuver duration. A perhaps more interesting and useful definition for the set $\mathcal{Q}$ is as a *safe* set of maneuvers; in the case of moving obstacles the set $\mathcal{Q}$ will be time dependent. In general, the computation of such a set is very challenging. In the hybrid system literature an approach that has been applied to several problems, including air traffic control, is based on the

definition and the computation of the level sets of an appropriately defined Hamiltonian function [28, 29]. A possibly conservative computation of the safe set can however be carried out quite easily in several cases; for example, in the case of fixed obstacles, each trajectory the execution of which allows for a complete stop along a collision free trajectory is safe.

Notice that the set $\mathcal{Q}$ is considerably smaller than the complete state space X; moreover, the hybrid control space V is typically smaller that the continuous control space U. This means that in general, solution to optimal control problems for the hybrid automaton will be computationally less expensive than solutions on the full state space. We found that the hybrid automaton model, for its structure and properties, lends itself in a very straightforward manner to a new class of motion planning algorithm, based on a randomized approach [30, 31, 32]. In particular, we will use a form of the so-called Rapidly-exploring Random Trees (RRT's), introduced in [33, 22].

Consider the case in which we want to control our system to the state $(\bar{q}, \bar{h}) \in \mathcal{Q}$, starting from the state $(q_0, h_0) \in \mathcal{Q}$. A concise statement of our version of the RRT algorithm is in the following.

1. We start building a tree, and assign (q_0, h_0) as its root. Also, set the index i to 0.
2. As the next step, we compute the optimal path in the obstacle free case: this will generate a sequence of states $\{(q_{i,1}, h_{i,1}), \ldots, (q_{i,n}, h_{i,n})\}$, where $q_{i,n} = \bar{q}$ and $h_{i,n}$ can be made arbitrarily close to $\bar{h}$;
3. define $\hat{n}$ as the index of the last one of the above states that belongs to the set $\mathcal{Q}$; if $\hat{n} = n$ then we have found a feasible solution to the problem. Moreover, if $i = 0$, the solution is the optimal one (subject to the dynamics represented in the trajectory primitives library), and we can exit. Otherwise add all the states $\{(q_{i,1}, h_{i,1}), \ldots, (q_{i,\hat{n}}, h_{i,\hat{n}})\}$ as a child to the node i;
4. choose (at random) a new "destination state" $(\tilde{q}, \tilde{h}) \in \mathcal{Q}$, and compute the corresponding optimal obstacle-free path, starting from the "closest" state in the tree (indexed as j). In this case, we can use as a distance function the optimal cost function $J^*(q, h \cdot \bar{h}^{-1})$. If the first state in the computed path is in $\mathcal{Q}$, then add it to the tree as a child of the state j, and set i to the index of the newly added tree node
5. if the maximum iteration number has not been exceeded, set $i \leftarrow i + 1$ and go back to step 2, otherwise exit with a failure.

The hybrid automaton architecture solves many of the issues that could pose a problem to the effectiveness of the RRT algorithm as presented in [33]. First of all, as stated above, the optimal cost function J^* represents a valid distance function, with a meaning that is directly related to the performance measure that we associate to the system (e.g. time to target in a minimum time problem). Moreover, once the optimal cost function is computed, the selection of the optimal controller and hence of the optimal path can be carried out in an extremely efficient and computationally inexpensive fashion. The need for a second tree, growing from the target point, is removed by the fact that at every step the tree grows in the direction of $(\bar{q}, \bar{h})$. This is in some sense reminiscent of the CONNECT step in [34]. For an online implementation, the tree growth is not parametrized with constant time steps that have to be kept "small", but instead with the coasting times (in trim trajectories) and the maneuver durations, that are not necessarily "small". This means that the computation of the new hybrid control input value is not required at a high rate.

The RRT algorithm is not complete, in the sense that it could fail to generate a feasible trajectory, even if one exists. However, it can be shown that the RRT algorithm provides a feasible solution with high probability [34]. Randomized algorithms have been found to perform very well in many applications of interest; however, RRTs cannot deal very well with workspace configurations that present "narrow passages" [35]. In many cases of interest though, like in the case of aerial vehicles, the environment is such that the work space does not have such narrow passages; moreover, in cases where this narrow passages exist, they could be known a priori, and appropriate way-points could be set.

4 Case study: control of an autonomous helicopter

A helicopter is a very good example of the systems for which we propose our control architecture. Helicopters are capable of remarkably agile and aggressive maneuvers, which are impossible to perform using traditional control techniques, especially when the on-line solution of the motion planning problem is required. Among the reasons for that we can state that helicopters are essentially unstable systems, with a very high bandwidth, and their dynamics change considerably throughout the flight envelope.

Several techniques for autonomous helicopter control have been proposed, ranging from robust linear control, neural control, fuzzy control, and nonlinear control [36, 37, 38]. We will focus on the latter techniques, mainly because of the fact that, if we can find control Lyapunov functions for the system, we can derive very easily estimates of the invariant sets R_q and Ω_q, as well as estimates of the image of a starting set C_q under a maneuver.

In recent papers, feedback linearization techniques have been applied to helicopter models, with positive results. The main difficulty in the application of such techniques is the fact that, for any significative selection of outputs, the helicopter dynamics are non-minimum phase, and hence are not directly input-output linearizable. However, it is possible to find good approximations to the helicopter dynamics such that the approximate system is input-output linearizable, and bounded tracking can be achieved [39, 19, 40]

The above mentioned approach suffers from a few drawbacks, one of the main ones being the fact that, since the attitude is parameterized using Euler angles, singularities arise when performing some maneuvers, such as loops, barrel rolls, split-s's etc. A possible solution to the singularity problem is represented by chart switching when approaching a singularity. However, this can be cumbersome in implementation, and can lead to excessively high gains in the proximity of singularities.

On the other hand, the singularities arising in these model are artifacts due to the choice of the attitude parameterization (Euler angles), and do not reflect any intrinsic characteristic of the helicopter dynamics. The need to avoid artificial singularities due the attitude representation is the main driver behind the control design presented in this report. To achieve this goal, we will operate directly in the configuration manifold of the helicopter.

The "tracking on manifolds" problem is solved in [41] for fully actuated mechanical systems. In the following we present an extension, for achieving asymptotic tracking of trajectories for a

particular class of underactuated mechanical systems. An approximation of the helicopter model can be shown to be in this class: the approximation that will be shown in the paper is the same one that leads to feedback linearizability, or differential flatness of the model. However, the method presented here can deal without any modification with more accurate models, including for example simple aerodynamic forces, a modeling of the tail side force.

4.1 Helicopter dynamics

The helicopter model that we will use here is based on the model presented in [19]. A very similar model has been widely used in the nonlinear control literature, in the three degrees of freedom case, as a VTOL aircraft model [39, 40]. More details on helicopter dynamics can be found in [42, 43, 44]. The dynamics of the helicopter as a rigid body can be expressed as in section (2.3), with the following expressions for the body force and moment in eq. (8):

$$\begin{cases} F_b &= mR^{-1}\tilde{g} + u_4 \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} + \begin{bmatrix} 0 & -\epsilon_2 & 0 \\ \epsilon_1 & 0 & -\epsilon_3 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 - \epsilon_4(u_4) \end{bmatrix} \\ M_b &= \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \end{cases} \tag{17}$$

In the above, $\tilde{g}$ is the gravity acceleration, and we have defined:

$$\begin{cases} u_1 &:= -z_{mr} T b_1 \\ u_2 &:= -z_{mr} T a_1 \\ u_3 &:= k_0 + k_T T^{1.5} + x_{tr} F_t \\ u_4 &:= T \end{cases} \tag{18}$$

and:

$$\begin{cases} \epsilon_1 &:= -\frac{1}{z_{mr}} \\ \epsilon_2 &:= -\frac{1}{z_{mr}} \\ \epsilon_3 &:= -\frac{1}{x_{tr}} \\ \epsilon_4(T) &:= k_0 + k_T T^{1.5} \end{cases} \tag{19}$$

where T is the main rotor thrust, $a1, b_1$ are respectively the pitch and roll rotor flapping angles, F_t is the tail rotor thrust, $-z_{mr}$ and $-x_{tr}$ are the moment arms of the main and tail rotor with respect to the helicopter center of mass, and k_0, k_T are the coefficient in the approximate expression of the main rotor reaction torque.

4.2 Control Design

The objective of the feedback control law will be to track a reference feasible trajectory $(g_{ref}(t), \xi_{ref}(t))$. We will start by designing a controller for the approximate system obtained by setting $\epsilon_i = 0, i = 1 \dots 4$. This approximation simplifies the control design considerably: it can be shown that the resulting approximate system is differentially flat, and hence feedback linearizable. Even though we will not use a feedback linearization technique, the absence of unstable zero dynamics gives an insight on the nature and the advantages of the approximate system.

4.2.1 Translational dynamics

If we consider as translational coordinates the position and velocity in the inertial frame $(p, \dot{p}) = (p, Rv)$, the translational dynamics is composed of three double integrators in parallel, driven by the force input $\alpha(R, u) := RF_b(u)$.

Since the translational dynamics block is essentially linear, it is easy to design a Lyapunov function V_p and a control policy $\mathcal{K}_p$ such that if

$$\alpha(R, u_4) = \bar{\alpha}(p, \dot{p}) := \mathcal{K}_p(p, \dot{p}) - m\tilde{g} \tag{20}$$

then the translational dynamics is stable, that is $\dot{V}_p(p, \dot{p}) \leq -W(p, \dot{p})$, where W is a positive definite function. The above can be easily extended for tracking of a reference trajectory $p_{ref}(t)$, by adding the appropriate feed-forward terms.

For simplicity, we will assume the following form for the translational dynamics control law:

$$\mathcal{K}_x := F_{ff} - K_p(p - p_{ref}) - K_{\dot{p}}(\dot{p} - \dot{p}_{ref}) \tag{21}$$

where the constant gains $K_p, K_{\dot{p}}$ can be derived from standard linear control design techniques, and F_{ff} is the feed-forward term.

If the function $\alpha(R, u_4)$ were invertible, then we would be able to use the attitude as a control input to the translational block. Unfortunately, this is not the case; however, we can select the desired attitude R_d as the "closest" element in $SO(3)$ (in the sense explained below) for which we can find a u_4 such that $\alpha(R_d, u_4) = \bar{\alpha}(p, \dot{p})$.

A measure of distance between two elements R_1, R_2 of $SO(3)$ can be derived from the relative rotation $\delta R := R_1 R_2^{-1}$ (group error), that is still an element of $SO(3)$ [41]. All the elements of $SO(3)$ can be described by a fixed axis $\tilde{r}$, corresponding to the single real eigenvector, with eigenvalue 1, and an angle of rotation $\tilde{\theta}$, which can be derived from the complex conjugate eigenvalues. As a measure of the magnitude of the group error δR, that is the distance between the rotations R_1 and R_2, we can consider the following function:

$$\Theta(\delta R) := 1 - \cos(\tilde{\theta}) = 2\sin^2\frac{\tilde{\theta}}{2} = \frac{1}{2}Tr(I - \delta R) \tag{22}$$

At this point, we are able to find the desired attitude and thrust by solving for the following optimization problem:

$$\begin{aligned} (R_d, u_4) \quad = \quad & \underset{(R,u)\in SO(3)\times\mathbb{R}}{\arg\min} \quad \Theta(RR_{ref}^{-1}) \\ & s.t. \quad \alpha(R, u) = \bar{\alpha}(p, \dot{p}) \end{aligned} \tag{23}$$

It can be verified that a unique solution exists, and that the dependence of R_d as defined above on $(p, \dot{p})$ is smooth, excluding the sets over which $R_{ref}e_3 \cdot \bar{\alpha}(p, \dot{p}) = 0$. This includes the case in which the commanded acceleration of the helicopter is equal to the gravity acceleration. This singularity is inherent to the physics of the problem, and as such cannot be avoided: it corresponds to the fact that if $||F_b|| = 0$ the helicopters enters a free fall, regardless of the attitude. Moreover,

in the case in which the commanded acceleration requires a rotation of $\pi/2$ radians of amplitude, there are two equivalent solutions to the problem (23), corresponding to $u_4 = \pm\bar{u}$.

As a matter of fact, the solution to the above minimization problem is given by:

$$R_d = \begin{cases} \text{Rot}\left(-R_{ref}e_3 \times \frac{\bar{\alpha}(p,\dot{p})}{||\bar{\alpha}(p,\dot{p})||_2},\ \cos^{-1}\left(-R_{ref}e_3 \cdot \frac{\bar{\alpha}(p,\dot{p})}{||\bar{\alpha}(p,\dot{p})||_2}\right)\right) R_{ref}, & \text{if } R_{ref}e_3 \cdot \bar{\alpha}(p,\dot{p}) \leq 0 \\ \text{Rot}\left(R_{ref}e_3 \times \frac{\bar{\alpha}(p,\dot{p})}{||\bar{\alpha}(p,\dot{p})||_2},\ \cos^{-1}\left(R_{ref}e_3 \cdot \frac{\bar{\alpha}(p,\dot{p})}{||\bar{\alpha}(p,\dot{p})||_2}\right)\right) R_{ref}, & \text{if } R_{ref}e_3 \cdot \bar{\alpha}(p,\dot{p}) > 0 \end{cases} \tag{24}$$

where $e_3 := [0,0,1]^T$, $\text{Rot}(r,\theta)$ is the rotation about the fixed axis r through an angle θ, and $\cdot$ and $\times$ represent the scalar and cross products of vectors in $\mathbb{R}^3$. The rotation $\text{Rot}(r,\theta)$ is given by the Rodrigues' formula:

$$\text{Rot}(r,\theta) = I_3 + \sin\theta\hat{r} + (1-\cos\theta)\hat{r}^2 \tag{25}$$

4.2.2 Attitude dynamics and backstepping control design

Once we have the desired attitude, using backstepping ideas [45, 46, 47], we want to track R_d in such a way to stabilize the overall system. However, before we can go on with the control design, we have to take a look at the rate of change of the objects introduced in the previous section (see also the derivations in [41]). First of all, we have that $\dot{R}_{ref} = R_{ref}\hat{\omega}_{ref}$; accordingly, we can define $\hat{\omega}_d := R_d^T\dot{R}_d$, where $\dot{R}_d$ can be computed componentwise; furthermore, we have that the following equalities hold:

$$\begin{aligned} \tfrac{d}{dt}\Theta(RR_d^T) &= \sin\tilde{\theta}\tfrac{d\tilde{\theta}}{dt} = \\ &= \text{Skew}(R_d^TR)^\vee \cdot (\omega - \omega_d) = \text{Skew}(RR_d^T)^\vee \cdot R_d(\omega - \omega_d) = \\ &= \sin\tilde{\theta}\ \tilde{r}\cdot R_d(\omega-\omega_d) = \sin\tilde{\theta}\ \tilde{r}_d\cdot(\omega-\omega_d) \end{aligned} \tag{26}$$

where $\tilde{r}, \tilde{\theta}$ are respectively the fixed axis and rotation angle of the attitude error RR_d^T, obtained by:

$$\cos\tilde{\theta} = \frac{Tr(RR_d^T)-1}{2} \tag{27}$$

$$\sin\tilde{\theta}\tilde{r} = \text{Skew}(RR_d^T)^\vee \tag{28}$$

We can rewrite eq.(26) as:

$$\frac{d}{dt}\Theta(RR_d^T) = \nabla\Theta\cdot(\omega-\omega_d) \tag{29}$$

having defined:

$$\nabla\Theta := \sin(\tilde{\theta})\tilde{r}_d = \text{Skew}(R_d^TR)^\vee \tag{30}$$

Now we are ready to state the following result:

Proposition 4.1 (Asymptotic tracking for the approximate system) *Given a smooth state trajectory $x_{ref}(t) = (g_{ref}(t), \xi_{ref}(t))$ for a rigid body under the action of the forces in eq. (17), with $\epsilon_i = 0, i = 1 \dots 4$, there exists an (almost everywhere) smooth static state feedback in explicit form under which the system state $x(t)$ globally asymptotically tracks the reference trajectory $x_{ref}(t)$.*

Proof: We will prove the above statement by actually building a tracking control law. Define a candidate Lyapunov function by adding to V_p terms that opportunely penalize the attitude configuration and velocity errors. Such a candidate Lyapunov function is the following:

$$V = V_p(p, \dot{p}) + k_\theta \Theta(R R_d^T) + \frac{1}{2}||\eta||_2^2 \tag{31}$$

where:

$$\eta := \omega - \omega_d - \frac{||\bar{\alpha}(p, \dot{p})||_2}{k_\theta \cos\frac{\tilde{\theta}}{2}} R_d^T \left(\tilde{t} \times \nabla_{\dot{p}} V_p\right) \tag{32}$$

and:

$$\tilde{t} := \frac{(R + R_d)F_b(u_4)}{||(R + R_d)F_b(u_4)||_2} \tag{33}$$

note that for $\tilde{t}$ to be well defined, we need $\bar{\alpha}(p, \dot{p}) \neq 0$.

Computing the time derivative of V, with the definition of R_d and u_4 given in the previous section, we get:

$$\frac{dV}{dt} \leq -W_p(p, \dot{p}) + \nabla_{\dot{p}} V_p(p, \dot{p})((R - Rd)F_b(u_4)) + \nabla\Theta \cdot (\omega - \omega_d) + \eta\dot{\eta} \tag{34}$$

We can make the above negative semidefinite by imposing:

$$\dot{\eta} = -k_\eta \eta - k_\theta \nabla\Theta \tag{35}$$

noting that:

$$\begin{gathered} \nabla_{\dot{p}} V_p(p, \dot{p})(R - R_d)F_b(u_4) = 2\sin\tfrac{\tilde{\theta}}{2}|u_4|\nabla_{\dot{p}} V_p \cdot \tilde{r} \times \tilde{t} = \\ = \left(k_\theta \sin\tilde{\theta} R_d^T \tilde{r}\right) \cdot \left(\frac{||\bar{\alpha}(p,\dot{p})||_2}{k_\theta \cos\frac{\tilde{\theta}}{2}} R_d^T \left(\tilde{t} \times \nabla_{\dot{p}} V_p\right)\right) \end{gathered} \tag{36}$$

so that we get:

$$\frac{dV}{dt} \leq -W_p(p, \dot{p}) - k_\eta ||\eta||_2^2 \leq 0 \tag{37}$$

The time derivative along system trajectory of the Lyapunov function V is hence negative-semidefinite: asymptotic stability can be inferred from LaSalle's principle. Note that the function Θ shares on $SO(3)$ the properties of radially unbounded functions on $\mathbb{R}^n$ for the purpose of global stability. The control law will be smooth almost everywhere, that is for all conditions for which

$\tilde{\theta} \neq \pi/2$. The explicit expression for the control torques $\mathbf{u} = [u_1, u_2, u_3]^T$, and the control force u_4 will then be given by:

$$\begin{cases} \mathbf{u} &= \omega \times \mathbb{J}\omega + \mathbb{J}\left[-k_\eta \eta - k_\theta \nabla\Theta + \frac{d\omega_d}{dt} + \frac{d}{dt}\left(\frac{||\bar{\alpha}(p,\dot{p})||_2}{k_\theta \cos\frac{\tilde{\theta}}{2}} R_d^T \left(\tilde{t} \times \nabla_{\dot{p}} V_p\right)\right)\right] \\ u_4 &= -R_d e_3 \cdot \bar{\alpha}(p,\dot{p}) \end{cases} \tag{38}$$

■

To the authors' knowledge, the above control law is a new result, providing asymptotic tracking for a class of underactuated mechanical systems on SE(3). The class of systems for which the control law is applicable comprises systems subject to a body-fixed force, and three independent torque components. The main advantage of (38) is the absence of artificial singularities, deriving from attitude parameterizations, like Euler angles.

4.2.3 Tracking for the actual model

So far, we have been able to design a controller to achieve asymptotic tracking of a reference trajectory for the approximate system. The term that we have neglected in the approximation will appear as disturbances in the nominal model. A first question that arises is how will the controller designed for the approximate system behave for the actual system, and in the presence of bounded external forces $||F_e||_2 \leq \Delta_e$, like for example those ensuing from uncertainties in the aerodynamics, or from wind gusts. We want to analyze the robustness properties of the control law given in eq. (38), in order to construct a consistent automaton, as defined in section (2.8).

Assume that the reference trajectory $x_{ref}(t) = (g_{ref}(t), \xi_{ref}(t))$ is feasible for a rigid body under the forces in (17). Note that if we replace in eq. (24) the nominal thrust direction in body axes e_3 with the actual thrust direction $\tilde{e}_3 := e_3 - \mathcal{E}(\mathbf{u_{ref}})$, as derived from eq. (17), the control laws (38) will give exact tracking for initial conditions on the reference trajectory, and no external disturbances.

As a first step in our analysis we will consider trim trajectories. Since the unmodeled forces F_u are a function of the control, given in eq. (17), that is a smooth function of the states and the reference trajectory, we have that, in a compact set $\mathcal{R} := x \in X, V(x, x_{ref}) \leq \bar{V}$, we can characterize the effect of the neglected coupling as:

$$||F_u||_2 \leq \Delta_u + \epsilon_p ||P||_2 \tag{39}$$

having defined $P := [p - p_{ref}, \dot{p} - \dot{p}_{ref}]^T$.

If we assume that we can measure the acceleration of the vehicle (this is not an unreasonable assumption, since autonomous vehicles are usually equipped with accelerometers), we have the following:

Proposition 4.2 (Bounded tracking for trim trajectories) *Given a trim trajectory $x_{ref}(t) = (\bar{g}\exp(\eta_{ref}\, t), \bar{\xi}_{ref})$ for a rigid body under the action of the forces in eq. (17), there exist sufficiently small $\epsilon_p, \Delta = \Delta_u + \Delta_e$ such the control law defined in section (4.2.2) is such that for all initial conditions in $\mathcal{R}$ the state $x(t)$ achieves bounded tracking of the reference trajectory $x_{ref}(t)$.*

Proof: If we compute the time derivative of the Lyapunov function V under the effect of the disturbance forces, we get that:

$$\begin{array}{rcl} \dot{V} & \leq & -W(P, Pref) + \nabla V_{\dot{p}}(F_e + F_u) - k_\eta ||\eta||_2^2 \leq \\ & \leq & -\alpha ||P||_2^2 + \beta ||P|| \left(\Delta + \epsilon_p ||P||_2\right) = -(\alpha - \beta\epsilon_p)||P_2||^2 + \beta\Delta||P|| \end{array} \tag{40}$$

Defining the set:

$$\Omega = \left\{ x \in X \mid V(x, x_{ref}) < \gamma \left(\frac{\beta\Delta}{\alpha - \beta\epsilon_p} \right)^2 \right\} \tag{41}$$

where γ is such that $V_p(P) \leq \gamma ||P||_2^2$ in $\mathcal{R}$, we have that, if $\epsilon_p < \alpha/\beta$, $\dot{V}$ is negative semi-definite in the set $\mathcal{R} \setminus \Omega$. Note that this, to be of any significance, requires that Δ be small enough that $\Omega \subset \mathcal{R}$, that is $\gamma \left(\frac{\beta\Delta}{\alpha - \beta\epsilon_p} \right)^2 < \bar{V}$. ■

Proposition (4.2) can be used to characterize the recoverability and limit set estimates for each trim trajectory. Also note that a similar result in [19] assumes that the disturbance F_u is bounded a priori; however, since F_u is a function of the state in the closed-loop system, this would require some restrictions on the system state, that we give by limiting the validity of the result to the set $\mathcal{R}$.

Since we have found a Lyapunov function for the nonlinear system, we can use the same function to study the behavior of the system during a maneuver. In the general case though, the computation of the Poincaré map will have to be carried out by specific techniques. In this case, define the set $\mathcal{C} := x \in X, V(x, x_{ref}) \leq \bar{V}$, and assume that we can characterize the effect of the neglected coupling as:

$$||F_u||_2 \leq \Delta_u + \epsilon_p ||P||_2 \tag{42}$$

for $x \in \mathcal{C}$. Then, by the same procedure used earlier, we have that:

$$2\gamma ||P||_2 \frac{d}{dt} ||P||_2 \leq \dot{V} \leq -(\alpha - \beta\epsilon_p)||P||_2^2 + \beta\Delta||P||_2 \tag{43}$$

After the maneuver duration t_m, we have that the state will be contained in the set:

$$\mathcal{D} := \left\{ x \in X | V(x, x_{ref}) < \gamma \left(\left(\sqrt{\frac{\bar{V}}{\chi}} - \frac{\beta\Delta}{\alpha - \beta\epsilon_p} \right) \exp\left(-(\alpha - \beta\epsilon_p)t_m\right) + \frac{\beta\Delta}{\alpha - \beta\epsilon_p} \right)^2 \right\} \tag{44}$$

where χ is such that $V_p(P) \geq \gamma ||P||_2^2$ in $\mathcal{R}$. Notice that by extending the maneuver duration t_m we can make the measure of the set $\mathcal{D} \setminus \Omega$ arbitrarily small. The above characterization of the destination sets $\mathcal{D}$ given a starting set $\mathcal{C}$ can be used to design maneuvers. At this point we have the tools for constructing a consistent automaton as defined in section (2.8).

As a further note, we would like to mention that the performance bounds just derived are obtained by using the original controller designed for the translational dynamics. However, the form of the disturbance input is such that is satisfies a matching condition: as a consequence, a robust control policy $\mathcal{K}(x)$ can be designed in a relatively straightforward manner to achieve better bounds. This possibility will be not be explored here, and is left to future research.

4.3 Simulation examples

In this section we will show some simulations obtained for the tracking control law, for "aggressive" maneuvers that cannot be handled in a straightforward manner by controllers based on an Euler angles parametrization of the attitude.

The first example that we will show is a stabilization problem. The helicopter starts at hover from the position $p = [-5, 5, -5]^T$, and with a NE heading of 45 degrees, and we want to hover at the origin, heading due North.

In fig. 4 we show both the response of the nominal system and the actual system, with the ϵ_i sized after typical values for a model helicopter. As we can see, the application of the controller designed for the nominal system to the actual system gives very good results.

We would like to remark that the simulation gives exactly the same results if we start at inverted hover (of course the attitude will be different by a 180 degree rotation). A second example is about tracking of a trajectory that performs a transition to inverted flight. This maneuver, in the case in which continuous controls are required, has to go through the singularity at $T = u_4 = 0$, since the thrust will be positive (upwards in the body frame) in the initial condition, and negative (downwards in the body frame) in the inverted flight condition. In the example, the controller was shut off when $|R_d e_3 \tilde{\alpha}(p, \dot{p})|$ was smaller than a preset value T_{min}. Also, notice that the required flapping angles a_1, b_1 diverge as $T \to 0$ (see eq. (18)).

As a matter of fact, on helicopters the rotor thrust can be changed very quickly by just a step in the collective: this is due to the fact that the kinetic energy stored in the rotor can be used to provide very fast commanded thrust responses. Hence the requirement of having a continuous thrust history can be relaxed.

In the transition to inverted flight maneuver that we are considering, we are close to the singular condition only for a short period of time, during which we cannot guarantee that $\dot{V} < 0$. However, the increment ΔV between before and after the singularity can be assumed to be arbitrarily small (by for example reducing T_{min}).

Again, the simulation plots (fig. 5) show good tracking of the reference trajectory.

4.4 Time optimal control in a free environment

For this application example, we consider the minimum time optimal control problem, in an obstacle-free environment. We want to take a 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:

$$g(q, h) = \begin{cases} 0 & \text{for } (q, h) = (q_{hover}, [\bar{x}, \cdot]^T), ||\bar{x}|| < \epsilon \\ 1 & \text{otherwise} \end{cases} \tag{45}$$

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

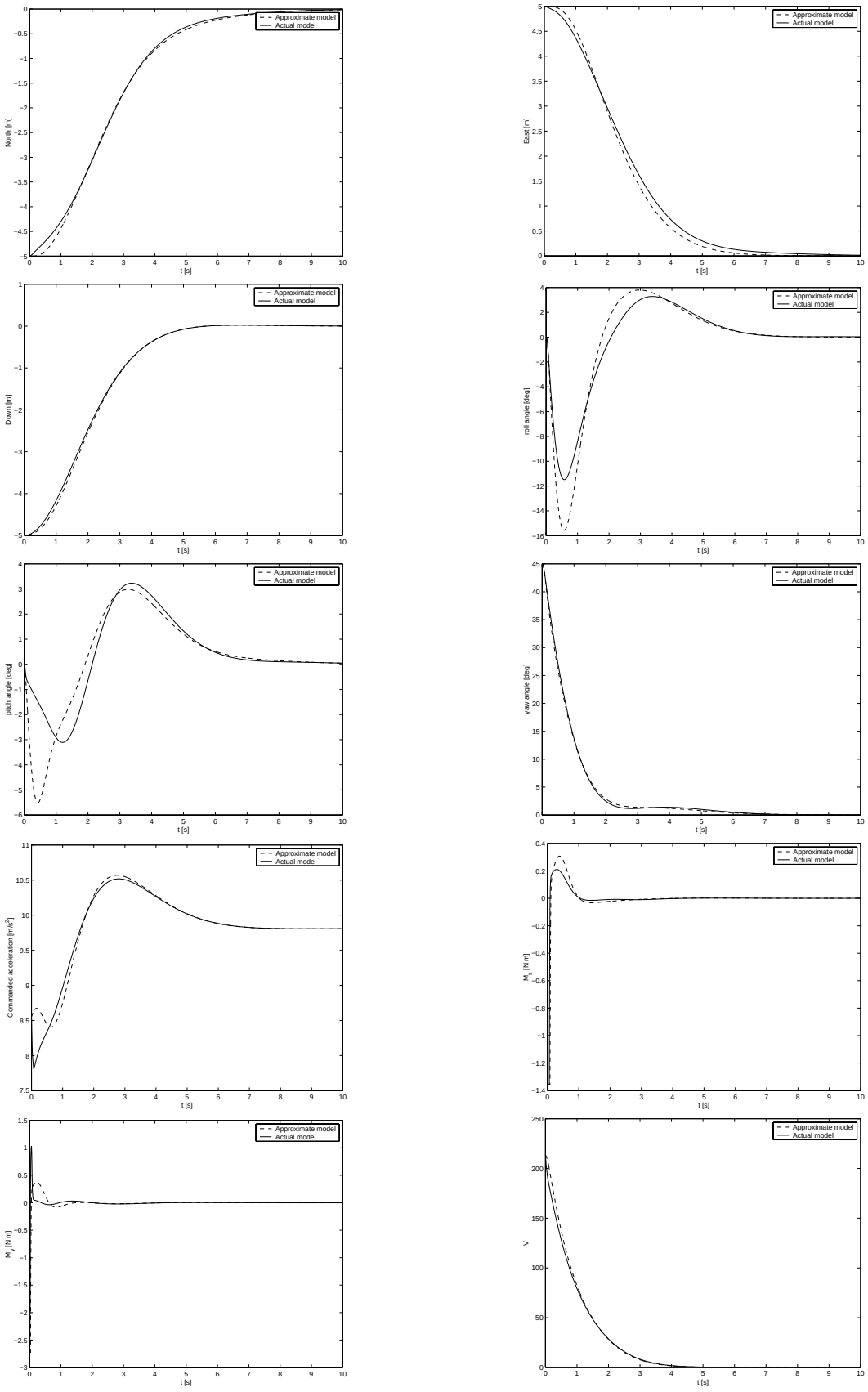

Figure 4: Simulation results for the first example: hover to the origin, with nonzero initial conditions

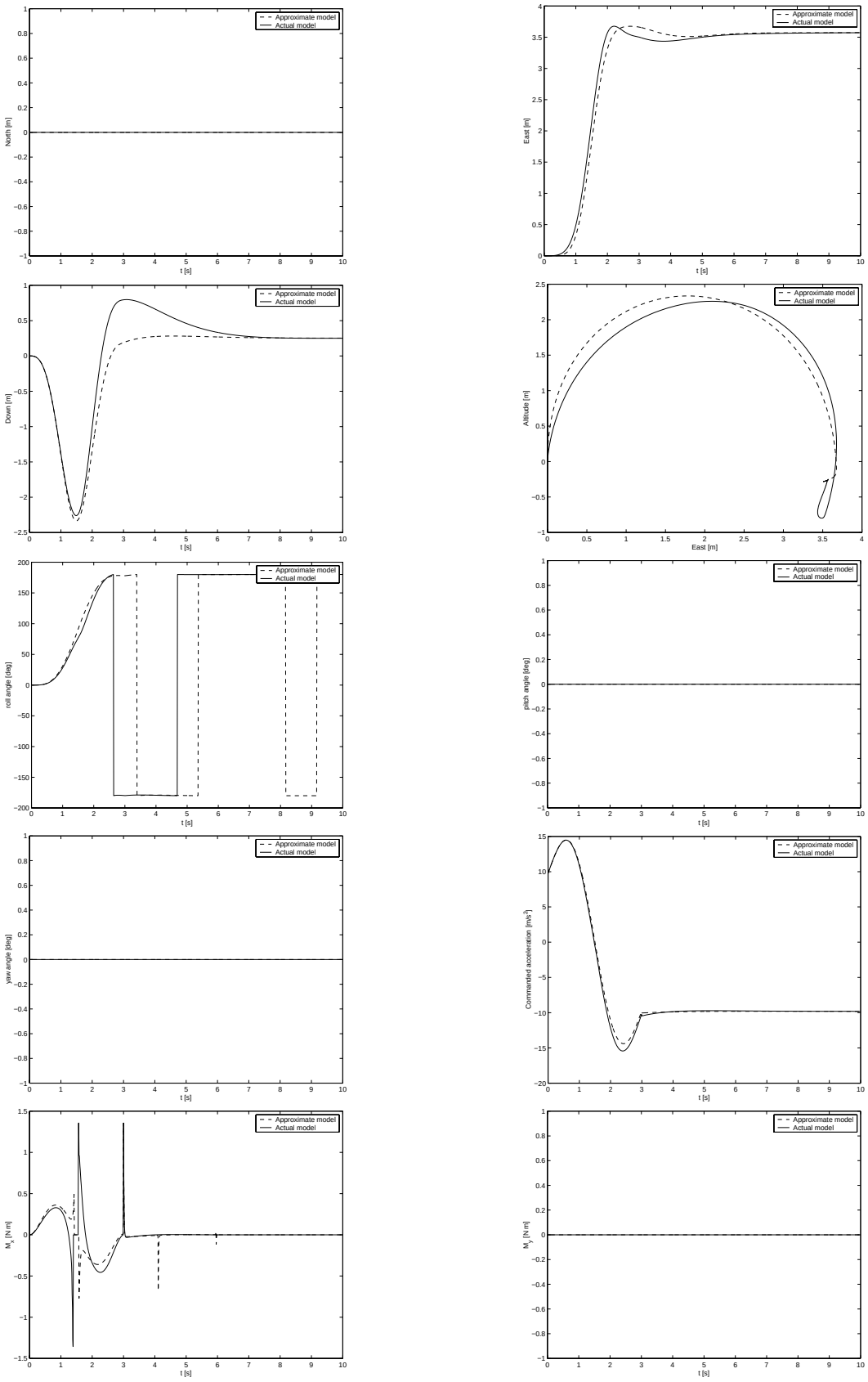

Figure 5: Simulation results for the second example: tracking of a maneuver (transition to inverted flight)

trajectory primitives), and computational issues (continuity at the optimum). As an alternative an additional terminal cost can be included in the cost expression.

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 (see fig. 6).

In the example, the design trim trajectories collection is defined by:

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

Reference maneuvers are computed for transition between all trim trajectories. Each maneuver is computed by connecting the trim parameters by splines of sufficiently high order to guarantee the required smoothness in the control inputs, and looking for the spline of least duration while satisfying constraints on the states, controls, and control rates.

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 application example, the computation of the optimal cost is carried out off-line (see fig. 7).

The evaluation of the optimal control, that is the computation of the minimizer (the arg min) of J as defined in eq. (16), which has to be done in real-time, 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. 8. In these figures, the height of the stems represents the velocity of the vehicle; moreover, solid lines and circle symbols indicate transitions in the hybrid automaton.

4.5 Motion planning with obstacles

As explained in section (3), the optimal cost function computed in the obstacle-free case can be profitably used in a randomized path-planning algorithm.

A variation of the algorithm summarized in section(3) has been implemented to allow for selection of the best feasible trajectory, still considering time as our performance measure. Essentially, such a modification consists in back-propagating the time to target each time a feasible child trajectory is added to the tree. By back-propagating we mean that we have to climb the tree back towards the root, labeling each node as the time to target of the child tree plus the time required for the transition from the node to the root of the child tree. This climbing process has to be reiterated until the label on the node under examination is actually smaller that the new computed value (this means that the current trajectory is not the best one found so far), or until we get to the maneuver tree root, in which case the current trajectory represents the best choice. Also, notice that care must be taken in labeling the nodes that belong to obstacle-free solutions as such, and

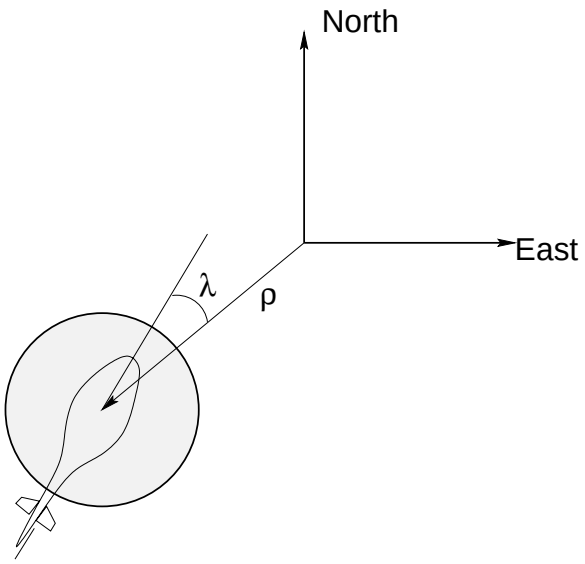

Figure 6: Example geometry

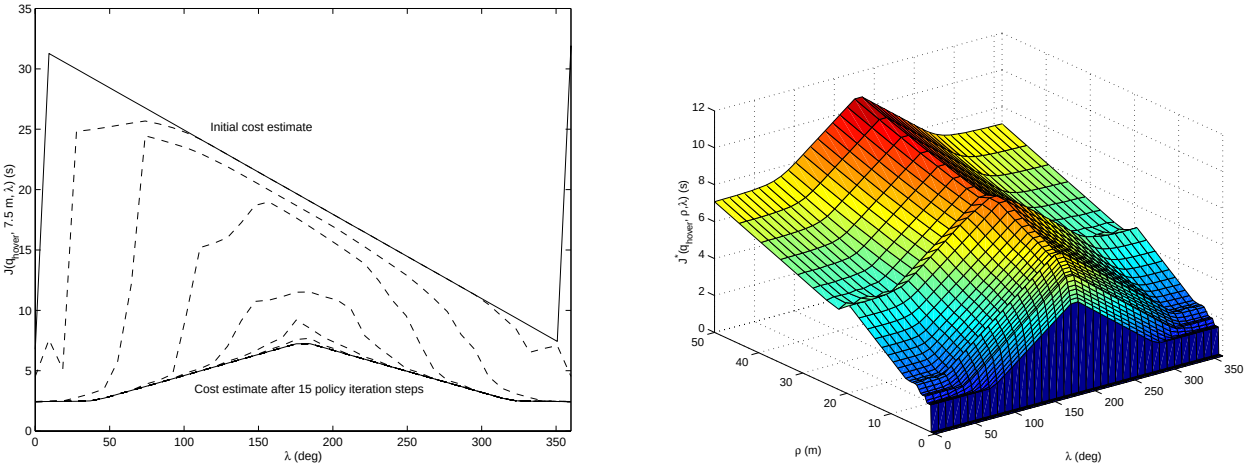

Figure 7: Value iteration results and optimal cost

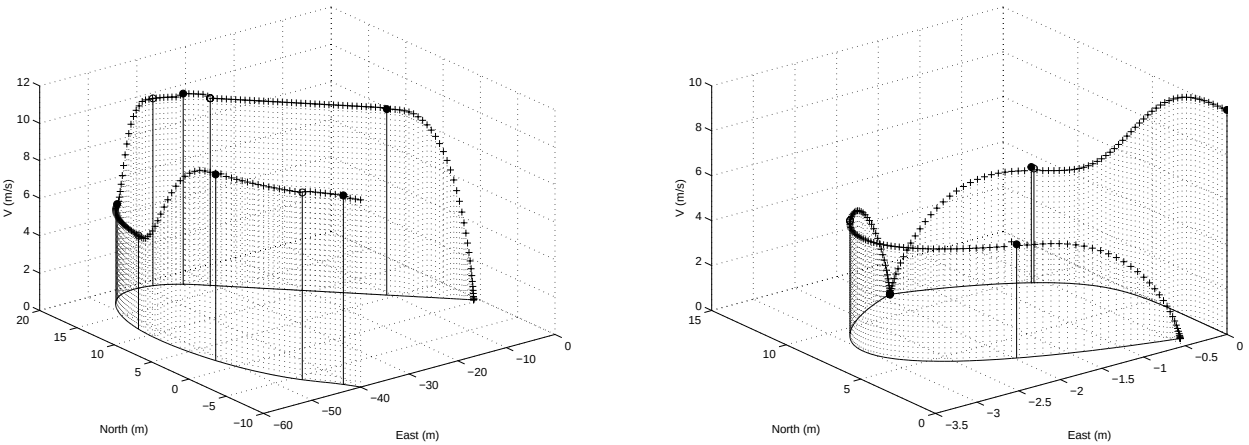

Figure 8: Simulated trajectory and velocity profile, starting from a high speed turn away from the target (left), and from high speed flight over the target

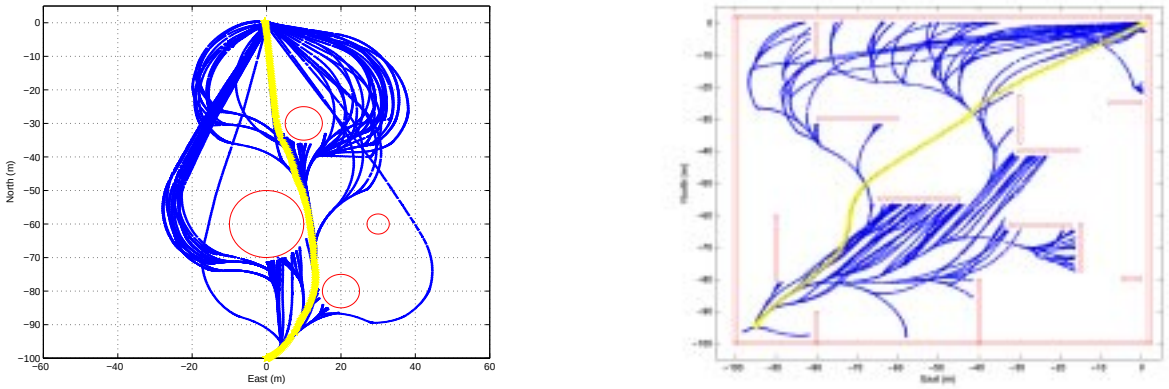

Figure 9: Simulated trajectory, and trace of the trajectory tree for two workspace configurations

exclude them from the randomized search. This is because an obstacle-free solution is optimal by construction, so we should not waste computing time in trying to improve it.

The randomized path planning has been tested in several examples, including cases with moving obstacles, and proved to be very fast and reliable. In the two cases depicted in fig. (9), the randomized motion planner succeeded in finding 50-100 feasible trajectories, for a running time ranging between 2 and 6 seconds.

5 Conclusions

In this report a Robust Hybrid Automaton architecture, applicable to autonomous vehicles has been presented and discussed. Algorithms have been given to solve, or approximate, the time-optimal motion planning problem in the free workspace case as well as in the presence of fixed and moving obstacles. A specific example, involving a small autonomous helicopter, has been presented in detail. The hybrid automaton structure has been found to provide a very flexible, and computationally effective tool for motion planning for autonomous vehicles. Current work includes the extension of the results in this report to real-time motion planning in an uncertain environment, and multi-vehicle operations.

6 Acknowledgments

The first author would like to thank Dr. Nicola Elia and Reza Olfati-Saber for insightful discussion, and Karen Sigurd for pointing out the RRT algorithm in [33] for motion planning. This research was supported by the C. S. Draper Laboratory through the IR&D grant DL-H-505334, by the AFOSR, under grant F49620-99-1-0320, and by the ONR, under Young Investigator Award N-00014-99-1-0668.

References

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

[2] J. Lygeros. *Hierarchical Hybrid Control of Large Scale systems.* PhD thesis, University of California, Berkeley, CA, 1996.

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

[4] N. Lynch, R. Segala, F. Vandraager, and H.B. Weinberg. Hybrid I/O automata. In R. Alur, T. Henzinger, and E. D. Sontag, editors, *Hybrid Systems III: Verification and Control*, volume 1066 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.

[5] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[6] T. Henzinger, P. H. Ho, and H. Wong-Toi. Algorithmic analsysis of nonlinear hybrid systems. *IEEE Trans. Aut. Control*, 43(4):540–554, 1998.

[7] 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.

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

[9] E. Frazzoli, M.A. Dahleh, and E. Feron. A hybrid control architecture for aggressive maneuvering of autonomous aerial vehicles. In *Advances in Systems Theory.* Kluwer Academic Publishers, 1999.

[10] J. E. Marsden and T.S. Ratiu. *Introduction to Mechanics and Symmetry.* Springer Verlag, New York, NY, 1994.

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

[12] B.L. Stevens and F.L. Lewis. *Aircraft Control and Simulation.* John Wiley & Sons, 1992.

[13] 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.

[14] R. R. Burridge, A. A. Rizzi, and D.E. Koditscheck. Sequential decomposition of dynamically dexterous robot behaviors. *International Journal of Robotics Research,*, 18(6):534–555, June 1999.

[15] M. Tittus and B. Egardt. Control design for integrator hybrid systems. *IEEE Trans. Aut. Control*, 43(4):491–500, April 1998.

[16] J.S. Shamma and J.R. Cloutier. Gain-scheduled missile autopilot design using linear parameter varying transformations. *AIAA J. on Guidance, Control an Dynamics*, 16(2):256–263, March-April 1993.

[17] I. Kaminer, A. Pascoal, E. Hallberg, and C. Silvestre. Trajectory tracking for autonomous vehicles: An integrated approach to guidance and control. *AIAA Journal of Guidance, Control, and Dynamics*, 21(1):29–38, January-February 1998.

[18] 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.

[19] 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.

[20] C. Tomlin, J. Lygeros, and S. Sastry. Aerodynamic envelope protection using hybrid control. In *Proc. American Control Conf.*, volume 3, pages 1793–1796, 1998.

[21] J. Hauser and R. Hindman. Manoeuvre regulation from trajectory tracking: feedback linearizable systems. In *Nonlinear Control Systems Design 1995. A Postprint Volume from the 3rd IFAC Symposium*, volume 1, pages 269–274, Oxford, UK, 1996. Pergamon.

[22] S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Available at: `http://janowiec.cs.iastate.edu/~lavalle/papers/rrt.ps` at the time of writing, 1999.

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

[24] D. P. Bertsekas and J.T. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

[25] S. M. LaValle. Robot motion planning: A game-theoretic foundation. *Algorithmica, special issue on the "Algorithmic Foundations of Robotics"*, 2000. To appear.

[26] T. Lozano-Perez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, 32(2):108–120, 1983.

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

[28] C. Tomlin, J. Lygeros, and S. Sastry. Synthesizing controllers for nonlinear hybrid systems. In *Proceedings of Hybrid Systems: Computation and Control*, volume 1386 of *Lecture Notes in Computer Science*. Springer-Verlag, 1998.

[29] C. Tomlin, G. J. Pappas, and S. Sastry. Conflict resolution for air traffic management: A study in multi-agent hybrid systems. *IEEE Trans. Aut. Control*, 43(4), April 1998.

[30] L.E. Kavraki, P. Svestka, J.C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[31] L. E. Kavraki, M. N Kolountzakis, and J.C. Latombe. Analysis of probabilistic roadmaps for path planning. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 3020–3025, 1996.

[32] L.E. Kavraki and J.C. Latombe. Probabilistic roadmaps for robot path planning. In K Gupta and A del Pobil, editors, *Practical Motion Planning in Robotics: Current Approaches and Future Directions*, pages 33–53. John Wiley, 1998.

[33] S.M. LaValle and J.J Kuffner. Randomized kinodynamic planning. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, 1999.

[34] J.J. Kuffner and S.M. LaValle. RRT-Connect: an efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, 2000. submitted.

[35] D. Hsu, L.E. Kavraki, J.C. Latombe, R. Motwani, and S. Sorkin. On finding narrow passages with probabilistic roadmap planners. In *Proceedings of the 1998 Workshop on Algorithmic Foundations of Robotics*, Houston, TX, March 1998.

[36] 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.

[37] J. Leitner, A. Calise, and J.V.R. Prasad. A full authority helicopter adaptive neuro-controller. In *IEEE Aerospace Conference Proceedings*, volume 2, pages 117–126, 1998.

[38] 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.

[39] J. Hauser, S. Sastry, and G. Meyer. Nonlinear control design for slightly nonminimum phase system: Application to v/stol aircraft. *Automatica*, 28(4):665–679, 1992.

[40] S. Sastry. *Nonlinear Systems: Analysis, Stability, and Control*, volume 10 of *IAM*. Springer Verlag, New York, NY, 1999.

[41] F. Bullo. *Nonlinear Control of Mechanical Systems: A Riemannian Geometry Approach*. PhD thesis, California Institute of Technology, 1998.

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

[43] R.W. Prouty. *Helicopter performance, stability, and control*. Krieger Pub. Co., 1990.

[44] G.D. Padfield. *Helicopter flight dynamics : the theory and application of flying qualities and simulation modeling*. American Institute of Aeronautics and Astronautics, 1996.

[45] M. Krstic, I. Kanellakopoulos, and P. Kokotovic. *Nonlinear and Adaptive Control Design*. John Wiley & Sons, 1995.

[46] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, 1996.

[47] R. Olfati-Saber. Normal forms for underactuated mechanical systems. Submitted to the American Control Conference, 2000.