Proceedings of the 42nd IEEE
Conference on Decision and Control
Maui, Hawaii USA, December 2003

Hierarchical Trajectory Generation for a Class of Nonlinear Systems[1]

Paulo Tabuada
Dept. of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556
e-mail: ptabuada@nd.edu

George J. Pappas
Dept. of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA 19104
e-mail: pappasg@seas.upenn.edu

Abstract

Trajectory generation and motion planning for nonlinear control systems is an important and difficult problem. In this paper, we provide a constructive method for hierarchical trajectory generation and hierarchical motion planning. The approach is based on the recent notion of ϕ-related control systems. Given a control affine system satisfying certain assumptions, we project the trajectory planning problem onto a ϕ-related control system of smaller dimension. Trajectories designed for the smaller, abstracted system are guaranteed, by construction, to be feasible for the original system. Constructive procedures are provided for refining trajectories of the coarser system to the more detailed system.

1 Introduction

Motion planning and trajectory generation for classes of nonlinear control systems has received a great deal of attention in the past decade. This has resulted in various motion planning approaches for nonholonomic systems [8] as well as real-time trajectory generation methods [14] for differentially flat systems. The rapidly growing interest in unmanned aerial vehicles (UAVs) has also emphasized the need to generate aggressive trajectories for individual UAVs ([4, 5]) as well as large numbers of autonomous UAVs.

Generating trajectories for complex nonlinear systems as well as generating trajectories for large numbers of interconnected nonlinear (UAV) systems naturally guide us towards a hierarchical approach to motion planning and trajectory generation. In this paper, we present such a hierarchical approach to motion planing and trajectory generation for nonlinear systems. The proposed methodology builds upon the notion of ϕ-related systems described in [10]. Given a control system Σ_1 with state space X_1, and a map $\phi : X_1 \longrightarrow X_2$, a ϕ-related system is an abstracted control system Σ_2 on the smaller state space X_2, that captures the ϕ-image of all Σ_1 trajectories. A construction is provided in [10] which given nonlinear model Σ_1 and map ϕ, generates the abstracted model Σ_2. Furthermore, given control theoretic properties such as controllability and stabilizability, we can obtain natural conditions on the map ϕ in order for Σ_1 and Σ_2 to have equivalent properties. Several important control theoretic properties such as controllability for nonlinear [10], and Hamiltonian systems [13] have been addressed, as well as stabilizability of linear systems [9].

This paper presents our approach to the following problem: Given a trajectory of the abstracted model Σ_2, we would like to refine this trajectory to a trajectory of the original control system Σ_1. For example, given a mechanical system, one would like to do motion planing for a kinematic model and later refine the planned trajectory to the full dynamical model. A solution to the above problem provides a hierarchical approach to motion planning, since we can transfer motion planning problems from Σ_1 to Σ_2, solve the motion planning problem on the simpler model Σ_2 using any existing method, and then refine the trajectory back to Σ_1. The explicit construction of this approach along with conditions that guarantee feasibility is the main contribution of this paper. The solution relies critically on our understanding of how state/input trajectories of the detailed system relate to the state/input trajectories of its abstraction. Such relations are described in [12] for fully nonlinear systems.

The idea of reducing the synthesis of control systems to simpler, lower dimensional systems has appeared in various forms in the literature. For mechanical systems, one such approach is based on the existence of symmetries, which allows to reduce a given control system to a simpler quotient system [3]. Recently, a different approach has been reported in [2], where kinematic models of mechanical systems, called kinematic reductions,

[1]This research is partially supported by NSF Information Technology Research Grant CCR01-21431, and NSF CAREER CCR-01-32716.

0-7803-7924-1/03/$17.00 ©2003 IEEE

generating trajectories that can be refined to trajectories of the full dynamical model are introduced. In the same spirit, the so-called inclusion principle [11] allows to carry analysis and design of systems to simpler models. Trajectory morphing [6] is a homotopy based approach that is essentially hierarchical. Backstepping has been a very successful approach for the recursive (or hierarchical) design of stabilizing controllers for nonlinear systems [7]. Backstepping is much closer in spirit to our approach but the focus in this paper is motion planning, not controller design.

2 ϕ-Related Control Systems

In this section, we review but also specialize the results in [10] to the class of nonlinear systems that we consider in this paper. We will consider control systems defined on $\mathbb{R}^m$, nevertheless some geometric notions will prove very useful. We follow the notation of [1]. We say that a given object is smooth when it is infinitely differentiable. Given a smooth map $\phi : \mathbb{R}^m \to \mathbb{R}^n$, we say that ϕ is a submersion when its associated tangent map $T\phi$ is surjective for every $x \in \mathbb{R}^m$. Given two vector fields $X : \mathbb{R}^m \to \mathbb{R}^m$ and $Y : \mathbb{R}^m \to \mathbb{R}^m$, we will denote by $[X, Y] : \mathbb{R}^m \to \mathbb{R}^m$ their Lie bracket defined by

$$[X,Y] = \frac{\partial Y_i}{\partial x_j} X_j - \frac{\partial X_i}{\partial x_j} Y_j$$

When S_1 and S_2 are sets of vector fields, $[S_1, S_2]$ will denote the set of vector fields defined as

$$[S_1, S_2] = \{X \ : \ \exists Y \in S_1, Z \in S_2 \text{ with } X = [Y, Z]\}$$

In this paper, we shall consider control systems which are affine in the control input.

Definition 2.1 *A control affine system* $\Sigma_1 = (\mathbb{R}^m, \mathbb{R}^k, F_1)$ *consists of state space* $\mathbb{R}^m$, *input space* $\mathbb{R}^k$, *and system map* $F_1 : \mathbb{R}^m \times \mathbb{R}^k \to \mathbb{R}^m$ *of the form*

$$F_1 = X_1 + \sum_{i=1}^{k} Y_1^i u_1^i$$

where $X_1, Y_1^1, \ldots, Y_1^k$ *are smooth vector fields on* $\mathbb{R}^m$ *and* $Y_1^1, \ldots, Y_1^k$ *are linearly independent.*

Given control affine system $\Sigma_1 = (\mathbb{R}^m, \mathbb{R}^k, F_1)$, it is natural to associate with Σ_1 the affine distribution

$$A_1 = X_1 + \text{span}\{Y_1^1, \ldots, Y_1^k\}$$

Trajectories of affine control systems are defined as follows:

Definition 2.2 *Let* $\Sigma_1 = (\mathbb{R}^m, \mathbb{R}^k, F_1)$ *be a control affine system and* $I \subseteq \mathbb{R}$ *an open interval containing the origin. A smooth curve* $x_1 : I \to \mathbb{R}^m$ *is said to be a state trajectory if there exists a (not necessarily smooth) input trajectory* $u_1 : I \to \mathbb{R}^k$ *satisfying the differential equation*

$$\dot{x}_1(t) = F_1(x_1(t), u_1(t))$$

for almost all $t \in I$.

We are interested in relating the trajectories of two models, possibly of different dimension. This is provided by the notion of ϕ-related control systems:

Definition 2.3 (ϕ-related control systems [10]) *Let* $\Sigma_1 = (\mathbb{R}^m, \mathbb{R}^k, F_1)$ *and* $\Sigma_2 = (\mathbb{R}^n, \mathbb{R}^l, F_2)$ *be control systems where* $n \leq m$ *and let* $\phi : \mathbb{R}^m \to \mathbb{R}^n$ *be a surjective submersion. Control system* Σ_2 *is said* ϕ*-related to* Σ_1 *if for every* $x_1 \in \mathbb{R}^m$:

$$T_{x_1}\phi\big(A_1(x_1)\big) \subseteq A_2(\phi(x_1)) \tag{2.1}$$

The notion of ϕ-related control systems allows us to relate the trajectories of the two control systems.

Theorem 2.4 ([10]) *Control system* Σ_2 *is* ϕ*-related to control system* Σ_1 *if and only if for every trajectory* $x_1(t)$ *of* Σ_1, $\phi(x_1(t))$ *is a trajectory of* Σ_2.

Even though Σ_2 captures the ϕ-image of every trajectory of Σ_1, it may also generate trajectories that are not feasible for the Σ_1 model. The goal of this paper is to reverse the direction of the above theorem, and hence refine trajectories of the coarser model Σ_2 to trajectories of the more detailed model Σ_1. This frequently occurs when, for example, trajectories of kinematic models must be refined to trajectories of dynamic models. In particular, in this paper, we shall address the following two problems.

Problem 2.5 (Hierarchical Trajectory Generation) *Let* Σ_2 *be a control system that is* ϕ*-related to control system* Σ_1. *Given a state trajectory* $x_2(t)$ *of* Σ_2 *corresponding to input trajectory* $u_2(t)$, *construct an input trajectory* $u_1(t)$ *for* Σ_1 *such that the resulting state trajectory* $x_1(t)$ *satisfies the relation* $\phi \circ x_1(t) = x_2(t)$.

Problem 2.6 (Hierarchical Motion Planning) *Let* Σ_2 *be a control system that is* ϕ*-related to control system* Σ_1. *Consider desired initial and final states* x_1^0, $x_1^F \in \mathbb{R}^m$ *for system* Σ_1. *Given a state trajectory* $x_2(t)$ *of* Σ_2 *satisfying* $x_2(0) = \phi(x_1^0)$ *and* $x_2(T) = \phi(x_1^F)$ *for some time* $T \in \mathbb{R}^+$, *construct an input trajectory* $u_1(t)$ *for* Σ_1 *such that the resulting state trajectory* $x_1(t)$ *satisfies* $\phi \circ x_1(t) = x_2(t)$, $x_1(0) = x_1^0$ *and* $x_1(T) = x_1^F$.

Even if Σ_2 is ϕ-related to Σ_1, Σ_2 may generate trajectories that not feasible for Σ_1. Hence extra conditions must be imposed on Theorem 2.3 in order to solve the above problems. In order to describe the solution to the previous problems we make the following assumptions for Σ_1 that will hold throughout the paper:

A.I: Control system Σ_2 is π-related to control system Σ_1 by the canonical projection $\phi = \pi : \mathbb{R}^m \to \mathbb{R}^n$, $\pi(x_1, \ldots, x_n, x_{n+1}, \ldots, x_m) = (x_1, \ldots, x_n)$

A.II: Control system Σ_1 is of the form

$$F_1 = X_1 + \sum_{i=1}^{a} Y_1^i u_1^i + \sum_{i=a+1}^{k} Y_1^i v_1^i$$

where $\ker(T\pi) = \text{span}\{Y_1^1, \ldots, Y_1^a\}$.

A.III: $\ker(T\pi) \subseteq \text{span}\{Y_1^1, \ldots, Y_1^k\}$.

A.IV: Affine distribution $A_1 = X_1 + \text{span}\{Y_1^1, \ldots, Y_1^a\}$ satisfies

$$\big[\big[A_1, \ker(T\pi)\big], \ker(T\pi)\big] \subseteq \ker(T\pi)$$

Assumptions A.I and A.II simplify the presentation of forthcoming results and can be made (locally) without loss of generality. Locally, every surjective submersion is a canonical projection, up to a change of coordinates. Similarly, assumption A.II can be satisfied by properly designing an invertible feedback transformation. Note that assumption A.II decomposes the inputs in two sets: inputs that will be retained ($1 \le i \le a$), and inputs that will be ignored ($a+1 \le i \le k$).

Assumptions A.III and A.IV are the critical conditions that will enable hierarchical trajectory generation and hierarchical motion planning. Intuitively, assumption A.III requires states projected out in the abstraction process to be directly controlled. Assumption A.IV is a partial nilpotency requirement between the controlled system, and the abstracted directions. Intuitively, it requires that $T\pi(X_1)$ and $T\pi(Y_1^i)$ are affine functions of $x_{n+1}, \ldots, x_m$.

In [10], a construction is presented which given a control affine system Σ_1 and a map ϕ, generates control system Σ_2 which is ϕ-related to it. We now present a restricted version of the construction in [10] which takes into account assumptions A.I through A.IV. In what follows we denote an element $x_1 \in \mathbb{R}^m$ as $x_1 = (x_2, x_c)$, $x_2 \in \mathbb{R}^n$, $x_c \in \mathbb{R}^{m-n}$ with $\pi(x_1) = x_2$.

Definition 2.7 (Constructing π-related systems) *Given control system $\Sigma_1 = (\mathbb{R}^m, \mathbb{R}^k, F_1)$ and canonical projection $\pi : \mathbb{R}^m \to \mathbb{R}^n$ compute:*

1. $X_2(x_2) = T_{(x_2,0)}\pi(X_1(x_2,0)) = \pi(X_1(x_2,0))$
2. $Y_2^i(x_2) = T_{(x_2,0)}\pi(Y_1^i(x_2,0)) = \pi(Y_1^i(x_2,0))$ *for* $i = 1, 2, \ldots, a$
3. $Y_2^{a+j}(x_2) = T_{(x_2,0)}\pi\Big(\frac{\partial X_1}{\partial x_{n+j}}\big|_{(x_2,0)}\Big)$ $= \pi\Big(\frac{\partial X_1}{\partial x_{n+j}}\big|_{(x_2,0)}\Big)$ *for* $j = 1, 2, \ldots, m-n$
4. $Y_2^{ij}(x_2) = T_{(x_2,0)}\pi\Big(\frac{\partial Y_1^i}{\partial x_{n+j}}\big|_{(x_2,0)}\Big)$ $= \pi\Big(\frac{\partial Y_1^i}{\partial x_{n+j}}\big|_{(x_2,0)}\Big)$ *for* $i = 1, 2, \ldots, a$ *and* $j = 1, 2, \ldots, m-n$.

and define the abstracted system by:

$$F_2 = X_2 + \sum_{i=1}^{a} Y_2^i u_2^i + \sum_{j=1}^{m-n} Y_2^{a+j} v_2^j + \sum_{i=1,j=1}^{a,m-n} Y_2^{ij} w_2^{ij}$$

Control system Σ_2 obtained by the previous construction is guaranteed to be π-related to Σ_1 [10]. Furthermore, the following equations clarify the relationships between states and inputs of systems Σ_1 and Σ_2

$$\begin{aligned} \pi(x_1) &= x_2 \\ u_1^i &= u_2^i \qquad i = 1, \ldots, a \\ x_c^j &= v_2^j \qquad j = 1, \ldots, m-n \end{aligned}$$

Note the mixing of inputs and states in the last equation. Inputs w_2^{ij} of Σ_2 do not correspond to a state or an input in Σ_1, but capture Lie bracket information that will be important in the next section[1].

3 Hierarchical trajectory generation

In this section we address the trajectory refinement problem. We start with a very simple Lemma which is used in the proof of the main result.

Lemma 3.1 *Let Σ_1 and Σ_2 be control systems and let $x_1(t)$ be a state trajectory for Σ_1 corresponding to input trajectory $u_1(t)$. The state trajectory $x_2(t)$ of Σ_2 corresponding to input trajectory $u_2(t)$ satisfies $\pi \circ x_1(t) = x_2(t)$ if and only if*

$$\pi \circ x_1(0) = x_2(0) \tag{3.1}$$

$$T_{x_1(t)}\pi \cdot F_1(x_1(t), u_1(t)) = F_2(x_2(t), u_2(t)) \tag{3.2}$$

Proof: Assume that $\pi \circ x_1(t) = x_2(t)$ holds. Then by setting $t = 0$ we obtain (3.1) while by time differentiation we get (3.2). Conversely, assume that (3.2) holds. Noting that (3.2) is equivalent to:

$$T_{x_1(t)}\pi \cdot \dot{x}_1(t) = \dot{x}_2(t) \quad \Leftrightarrow \quad \frac{d}{dt}(\pi \circ x_1(t)) = \frac{d}{dt}x_2(t)$$

[1] The relationship between the inputs and states of Σ_1 and Σ_2 is described in more detail in [12].

we obtain that $\pi \circ x_1(t) = x_2(t) + c$ for some constant vector $c \in \mathbb{R}^n$. However, (3.1) implies that $c = 0$, so that we get $\pi \circ x_1(t) = x_2(t)$ as desired. ■

We now present a solution Problem 2.5. The heart of the proposed construction is the observation that control inputs v_2^j of coarser model Σ_2 can be identified with the ignored states x_c^j of detailed model Σ_1.

Theorem 3.2 (Hierarchical trajectory generation) *Let $\Sigma_1 = (\mathbb{R}^m, \mathbb{R}^k, F_1)$ be a control affine system satisfying assumptions* A.I *through* A.IV. *Let $\pi : \mathbb{R}^m \to \mathbb{R}^n$ be the canonical projection, and $\Sigma_2 = (\mathbb{R}^n, \mathbb{R}^l, F_2)$ the π-related control system obtained by construction described in Definition 2.7. Given any smooth state trajectory $x_2(t)$ of Σ_2 corresponding to any smooth input trajectory $(u_2(t), v_2(t), w_2(t))$ satisfying*

$$w_2^{ij}(t) = u_2^i(t)v_2^j(t) \quad i = 1, \ldots, a,\ j = 1, \ldots, m-n$$

there exists a trajectory $x_1(t)$ of Σ_1 satisfying $\pi \circ x_1(t) = x_2(t)$. Furthermore, the state trajectory $x_1(t)$ of Σ_1 is given by:

$$x_1(t) = (x_2(t), x_c(t)) = (x_2(t), v_2(t)) \tag{3.3}$$

and corresponds to input trajectory:

$$\begin{bmatrix} u_1^1(t) \\ \vdots \\ u_1^a(t) \\ u_1^{a+1} \\ \vdots \\ u_1^k \end{bmatrix} = \begin{bmatrix} u_2^1(t) \\ \vdots \\ u_2^a(t) \\ \alpha_{a+1}(t) \\ \vdots \\ \alpha_k(t) \end{bmatrix} \tag{3.4}$$

where $\alpha(t) = (\alpha_{a+1}(t), \ldots, \alpha_k(t))$ is given by the solution of:

$$\dot{v}_2(t) = PX_1 + P \sum_{i=a+1}^{k} Y_1^i \alpha_i(t) \tag{3.5}$$

with $P : \mathbb{R}^m \to \ker(T\pi) \cong \mathbb{R}^{m-n}$, the canonical projection mapping $P(x_1) = x_c$.

Remark: Before proving this result we note that (3.5) can always be solved for α. To see this, we construct the matrix $Y = [Y_1^{a+1} | Y_1^{a+2} | \ldots | Y_1^k] : \mathbb{R}^{m-n} \to \mathbb{R}^m$ and note that $PY : \mathbb{R}^{n-m} \to \ker(T\pi) \cong \mathbb{R}^{n-m}$ is an isomorphism. This follows from the fact that the vector fields Y_1^i for $i = a+1, a+2, \ldots, k$ form a basis for $\ker(T\pi)$. The input trajectories $\alpha(t)$ can then be obtained by:

$$\alpha(t) = (PY)^{-1}(\dot{v}_2(t) - PX_1)$$

Although the vector PX_1 and matrix PY are state dependent, we can express $x_1(t)$ as a function of $x_2(t)$ and $v_2(t)$ in virtue of (3.3) thereby defining α as a function of time.

Proof: The result follows from Lemma 3.1 provided that (3.1) and (3.2) hold. We start by showing that (3.2) is satisfied. Input trajectory $u_1(t) = (u_2(t), \alpha(t))$ defines state trajectory $x_1(t)$ by:

$$\begin{aligned}
\dot{x}_1(t) &= X_1(x_1(t)) + \sum_{i=1}^{a} Y_1^i u_2^i(t) + \sum_{a+1}^{k} Y_1^i \alpha^i(t) \\
&= X_1(x_2(t), 0) + \sum_{j=1}^{m-n} \left.\frac{\partial X_1}{\partial x_{n+j}}\right|_{(x_2(t),0)} x_{n+j}(t) \\
&+ \sum_{i=1}^{a} \Big(Y_1^i(x_2(t), 0) + \sum_{j=i}^{m-n} \left.\frac{\partial Y_1^i}{\partial x_{n+j}}\right|_{(x_2(t),0)} x_{n+j}(t) \Big) u_2^i(\\
&+ \sum_{i=a+1}^{k} Y_1^i \alpha^i(t)
\end{aligned}$$

where we have twice used the fact that any vector field $Z(x_1) = Z(x_2, x_c)$ satisfying assumption A.IV can be written as

$$Z(x_2, x_c) = Z(x_2, 0) + \sum_{j=1}^{m-n} \left.\frac{\partial Z}{\partial x_{n+j}}\right|_{(x_2,0)} x_{n+j}$$

Projecting on $\mathbb{R}^n$ using $T\pi$ results in:

$$\begin{aligned}
T_{x_1(t)}\pi(\dot{x}_1(t)) &= X_2(x_2(t)) \\
&+ \sum_{j=1}^{m-n} Y_2^j(x_2(t)) x_{n+j}(t) \\
&+ \sum_{i=1}^{a} Y_2^i u_2^i(t) \\
&+ \sum_{i=1,j=1}^{a,m-n} Y_2^{ij}(x_2(t)) x_{n+j}(t) u_1^i(t)
\end{aligned}$$

Since $\alpha(t)$ satisfies (3.5), it follows that $x_{n+j}(t) = v_2^j(t)$. Furthermore, by making use of the equality $w_2^{ij}(t) = u_2^i(u) v_2^j(t)$ we obtain:

$$\begin{aligned}
T_{x_1(t)}\pi(\dot{x}_1(t)) &= X_2(x_2(t)) \\
&+ \sum_{j=1}^{m-n} Y_2^j(x_2(t)) v_2^j(t) \\
&+ \sum_{i=1}^{a} Y_2^i u_2^i(t) \\
&+ \sum_{i=1,j=1}^{a,m-n} Y_2^{ij}(x_2(t)) w_2^{ij}(t) \\
&= F_2(x_2(t), u_2(t), v_2(t), w_2(t))
\end{aligned}$$

which shows that (3.2) is satisfied. We now see that if the initial condition $x_1(0)$ satisfies (3.1), which is always possible, then the trajectory $x_1(t)$ corresponding

to input trajectory $(u_2(t), \alpha(t))$ satisfies $\pi \circ x_1(t) = x_2(t)$ in virtue of Lemma 3.1. ■

Theorem 3.2 provides a constructive procedure for refining of state/input trajectories of Σ_2 to state/input trajectories of Σ_1. In addition to trajectory refinement, Theorem 3.2 can also be used for hierarchical motion planning, thus leading to the solution of Problem 2.6. Suppose we wish to determine a trajectory of control system Σ_1 connecting point $x_1^0 \in \mathbb{R}^m$ to point $x_1^F \in \mathbb{R}^m$. Then, Theorem 3.2 states that this reduces to finding a trajectory $x_2(t)$ for Σ_2 joining $\pi(x_1^0)$ to $\pi(x_1^F)$. However, the refinement process only ensures that trajectory $x_1(t)$ satisfies $\pi \circ x_1(t) = x_2(t)$ and additional assumptions are necessary to ensure that $x_1(t)$ links x_1^0 to x_1^F. Such additional requirements are summarized in the next corollary of Theorem 3.2:

Corollary 3.3 (Hierarchical Motion Planning) *Let $\Sigma_1 = (\mathbb{R}^m, \mathbb{R}^k, F_1)$ be a control affine system satisfying Assumptions* A.I *through* A.IV. *Let $\pi : \mathbb{R}^m \to \mathbb{R}^n$ be the canonical projection, and $\Sigma_2 = (\mathbb{R}^n, \mathbb{R}^l, F_2)$ the π-related control system obtained by construction described in Definition 2.7. Consider any two states $x_1^0 = (x_2^0, x_c^0)$ and $x_1^F = (x_2^F, x_c^F)$ in $\mathbb{R}^m$, and let $x_2(t)$ be any smooth state trajectory of Σ_2 such that $x_2(0) = x_2^0$ and $x_2(T) = x_2^F$ for some $T \in \mathbb{R}^+$. If $x_2(t)$ corresponds to a smooth input trajectory $(u_2(t), v_2(t), w_2(t))$ satisfying*

$$\begin{aligned} w_2^{ij}(t) &= u_2^i(t) v_2^j(t) \quad i = 1, \ldots, a,\ j = 1, \ldots, m-n \\ v_2(0) &= x_c^0 \\ v_2(T) &= x_c^F \end{aligned}$$

then the state trajectory $x_1(t)$ of Σ_1 described in Theorem 3.2 satisfies $x_1(0) = x_1^0$ and $x_1(T) = x_1^F$.

Proof: The input trajectory $u_1(t) = (u_2(t), \alpha(t))$ defined in Theorem 3.2 satisfies (3.5), which implies that $x_{n+j}(t) = v_2^j(t)$. We thus have that $x_1(0) = (x_2(0), x_c(0)) = (x_2^0, v_2(0)) = (x_2^0, x_c^0) = x_1^0$. Similarly one shows that $x_1(T) = x_1^F$ which concludes the proof. ■

Theorem 3.2 and Corollary 3.3 reflect the natural tradeoff that arises in hierarchical methods : ignoring some Σ_1 dynamics can be performed using this framework, as long as Σ_2 satisfies certain algebraic constraints. For example, the algebraic input constraints $w_2^{ij}(t) = u_2^i(t) v_2^j(t)$ at the level of Σ_2 accommodate certain Lie bracket conditions between ignored input vector fields of Σ_1. Algebraic constraints are clearly much more desirable computationally than differential constraints.

4 An illustrative example

Consider control system

$$F_1 = X_1 + Y_1^1 u_1^1 + Y_1^2 u_1^2$$

with state $(x_1, x_2, x_3) \in \mathbb{R}^3$ and inputs $u_1^1, u_1^2 \in \mathbb{R}$, and vector fields defined by:

$$X_1 = \begin{bmatrix} x_1(1 + x_2 + x_1 x_3) \\ x_1(x_1 + x_2) \\ x_1^2 x_2 \end{bmatrix} \tag{4.1}$$

$$Y_1^1 = \begin{bmatrix} 0 \\ x_3 \\ 0 \end{bmatrix} \quad Y_1^2 = \begin{bmatrix} 0 \\ 0 \\ x_1(x_1^2 + x_3) \end{bmatrix} \tag{4.2}$$

Our goal is to construct a state trajectory joining the states

$$\begin{bmatrix} x_1^0 \\ x_2^0 \\ x_3^0 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \\ 1/4 \end{bmatrix} \quad \begin{bmatrix} x_1^F \\ x_2^F \\ x_3^F \end{bmatrix} = \begin{bmatrix} 4 \\ -1 \\ 1/36 \end{bmatrix} \tag{4.3}$$

We proceed by choosing the projection map π to be $\pi(x_1, x_2, x_3) = x_1$, which satisfies:

$$\ker(T\pi) = \operatorname{span}\left\{ \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right\} = \operatorname{span}\left\{ Y_1^1, Y_1^2 \right\} \tag{4.4}$$

Hence assumptions A.II and A.III hold provided that $x_3 \neq 0$, $x_1 \neq 0$ and $x_1^2 \neq -x_3$. Assumption A.IV also holds as can be seen by computing the partial derivatives of vector fields (4.1) and (4.2) with respect to x_2 and x_3 (the ignored states).

We now follow the construction described in Definition 2.7 to obtain system Σ_2 which is π-related to system Σ_1. Construction 2.7 is significantly simplified since $a = 0$. Therefore steps 2 and 4 of the construction can be ignored since in this case we have $a = 0$ due to equality (4.4). This leaves us with steps 1 and 3 for the construction.

1.

$$\begin{aligned} X_2(x_1) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} X_1(x_1, 0, 0) \\ &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_1^2 \\ 0 \end{bmatrix} = x_1 \end{aligned}$$

3.

$$\begin{aligned} Y_2^{a+1} = Y_2^1 &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \left. \frac{\partial X_1}{\partial x_2} \right|_{(x_1, 0, 0)} \\ &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ 0 \\ x_1^2 \end{bmatrix} = x_1 \end{aligned}$$

$$Y_2^{a+2} = Y_2^2 = \begin{bmatrix}1 & 0 & 0\end{bmatrix} \frac{\partial X_1}{\partial x_3}\bigg|_{(x_1,0,0)}$$
$$= \begin{bmatrix}1 & 0 & 0\end{bmatrix} \begin{bmatrix}x_1^2 \\ x_1 \\ 0\end{bmatrix} = x_1^2$$

This results in control system Σ_2 defined :

$$\dot{x}_1 = F_2(x_1, v_2^1, v_2^2) = x_1 + x_1 v_2^1 + x_1^2 v_2^2$$

with state $x_1 \in \mathbb{R}$ and inputs v_2^1,$v_2^2 \in \mathbb{R}$. We must now find a Σ_2 trajectory connecting $x_1(0) = 2 = x_1^0$ and $x_1(2) = 4 = x_1^F$ (hence $T = 2$). One such trajectory can be obtained by simply canceling the drift term, by choosing $v_2^1 = -1$ and $v_2^2 = 1/x_1^2$, thus obtaining $\dot{x}_1 = 1$. The motion planning problem for Σ_2 is now trivial since if one starts at $x_1(0) = 2 = x_1^0$ we have $x_1(2) = 4 = x_1^F$. Furthermore $v_2^1(0) = -1 = v_2^2(2) = x_2(0) = x_2(2)$ and $v_2^2(t) = 1/((t+2)^2) \Rightarrow v_2^2(0) = 1/4 = x_3(0)$ and $v_2^2(2) = 1/36 = x_3(2)$ which shows that the conditions of Corollary 3.3 are indeed satisfied.

Our objective is now to refine Σ_2 trajectories to Σ_1 trajectory. Since $a = 0$, the input trajectory for Σ_1 is simply given by α, the solution of (3.5):

$$\begin{bmatrix}\dot{v}_2^1 \\ \dot{v}_2^2\end{bmatrix} = \begin{bmatrix}x_1(x_1 + x_2) \\ x_1^2 x_2\end{bmatrix} + \begin{bmatrix}x_3 & 0 \\ 0 & x_1(x_1^2 + x_3)\end{bmatrix} \begin{bmatrix}\alpha_1 \\ \alpha_2\end{bmatrix}$$

which can be solved for α_1,α_2 resulting in

$$\alpha_1 = \frac{-x_1(x_1 + x_2)}{x_3}$$
$$\alpha_2 = \frac{-2(1 + x_2 + x_1 x_3) - x_1^4 x_2}{x_1^3(x_1^2 + x_3)}$$

Expressing $(x_1(t), x_2(t), x_3(t))$ as functions of time using $x_1(t) = t + 2$, $v_2^1(t) = x_2(t) = -1$ and $v_2^2(t) = x_3(t) = 1/x_1^2(t) = 1/(t+2)^2$ we obtain:

$$\alpha_1(t) = (t+2)^3(t+1)$$
$$\alpha_2(t) = \frac{(t+2)^5 - 2}{(t+2)^2((t+2)^4 + 1)}$$

Corollary 3.3 now ensures that by using inputs $u_1^1(t) = \alpha_1(t)$ and $u_1^2(t) = a_2(t)$, and starting at $(x_1(0), x_2(0), x_3(0)) = (x_1^0, x_2^0, x_3^0)$, the trajectory $(x_1(t), x_2(t), x_3(t))$ of Σ_1 will satisfy $(x_1(T), x_2(T), x_3(T)) = (x_1^F, x_2^F, x_3^F)$.

5 Discussion

We have presented a constructive methodology for hierarchical trajectory generation and motion planning. Several assumptions made in this paper allowed for a simplified presentations of the results but are in fact not essential. Future work will relax the required assumptions and provide similar results in a more general setting. In particular assumption *A.V* can weakened at the expense of more algebraic constraints. Future work will also focus on the several interesting relations with similar work such as backstepping, flatness, and kinematic reductions.

References

[1] R. Abraham, J. Marsden, and T. Ratiu. *Manifolds, Tensor Analysis and Applications.* Applied Mathematical Sciences. Springer-Verlag, 1988.

[2] Francesco Bullo and Kevin M. Lynch. Kinematic controllability for decoupled trajectory planning in underactuated mechanical systems. *IEEE Transactions on Robotics and Automation*, 17(4):402–412, August 2001.

[3] G. S. de Alvarez. Controllability of poisson control systems with symmetries. In J. E. Marsden, P. S. Krishnaprasad, and J. C. Simo, editors, *Dynamics and Control of Multi-body Systems*, volume 97 of *Contemporary Mathematics*. A.M.S., 1989.

[4] E. Frazzoli, M. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. In *Proceedings of the 2001 American Control Conference*, pages 43–48, Arlington, VA, June 2001.

[5] J. Hauser and A. Jadbabaie. Aggressive maneuvering of a thrust vectored flying wing : A receding horizon approach. In *Proceedings of the 39th IEEE Conference on Decision and Control*, pages 3582–3587, Sydney, Australia, December 2000.

[6] J. Hauser and D.G. Meyer. Trajectory morphing for nonlinear systems. In *Proceedings of the 1998 American Control Conference*, pages 2065–2070, Philadelphia, PA, June 1998.

[7] M. Krstic, I. Kanellakopoulos, and P. Kokotovic. *Nonlinear and Adaptive Control Design.* Wiley, New York, NY, 1995.

[8] R.M. Murray and S. Sastry. Nonholonomic motion planning: Steering using sinusoids. *IEEE Transactions on Automatic Control*, 38(5):700–716, 1993.

[9] George J. Pappas and Gerardo Lafferriere. Hierarchies of stabilizability preserving linear systems. In *Proceedings of the 40th IEEE Conference on Decision and Control*, pages 2081–2086, Orlando, Florida, December 2001.

[10] George J. Pappas and Slobodan Simic. Consistent hierarchies of affine nonlinear systems. *IEEE Transactions on Automatic Control*, 47(5):745–756, 2002.

[11] Srdjan S. Stankovic and Dragoslav D. Siljak. Model abstraction and inclusion principle: A comparison. *IEEE Transactions on Automatic Control*, 47(3):529–532, 2002.

[12] Paulo Tabuada and George J. Pappas. Quotients of fully nonlinear control systems. *SIAM Journal on Control and Optimization*, 2001. Under revision, available at `www.seas.upenn.edu/~tabuadap`.

[13] Paulo Tabuada and George J. Pappas. Abstractions of Hamiltonian control systems. *Automatica*, 2002. To appear, available at `www.seas.upenn.edu/~tabuadap`.

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