Algorithms for Hybrid Optimal Control

Aude Rondepierre and Jean-Guillaume Dumas

Laboratoire de Modélisation et Calcul
Tour IRMA - BP 53, 38041 Grenoble cedex 9. France
Aude.Rondepierre@imag.fr
www-lmc.imag.fr/lmc-mosaic/Aude.Rondepierre

Abstract. We consider a non linear ordinary differential equation and want to control its behavior so that it reaches a target by minimizing a cost function. Our approach is to use hybrid systems to solve this problem: the complex dynamic is replaced by piecewise affine approximations which allow an analytical resolution. The sequence of affine models then forms a sequence of states of a hybrid automaton. Given an optimal sequence of states, we are then able to traverse the automaton till the target, locally insuring the optimality.

Keywords. Affine and Nonlinear Optimal Control Problems, Canonical Transformation, Controllability, Hybrid System.

Introduction

Aerospace engineering, automatics and other industries provide a lot of optimization problems, which can be described by optimal control formulations: change of satellites orbits, flight planning, motion coordination [11] ([22] for more applications in aerospace industry). Now, in "real-life", optimal control problems are fully nonlinear. Since the years 1950-1970, the theory of optimal control has been extensively developed and has provided us with powerful results like dynamic programming [2] or the maximum principle [24]. A large amount of theory have been well studied, but resolutions are mainly numerical.

In this paper, we consider a dynamical system which state is described by the solution of the following ordinary differential equation (ODE):

$$\left\{\begin{array}{l}\dot{X}(t) = f(X(t), u(t)) \\ X(0) = X_0\end{array}\right. \tag{1}$$

We present a hybrid algorithm controlling the system (1) from an initial state X_0 at time $t = 0$ to a final state $X_f = 0$ at an unspecified time t_f. To reach this state, we allow the admissible control functions u to take values in a convex and compact polyhedral set $\mathbb{U}_m$ of $\mathbb{R}^m$, in such a way that:

$$J(X, u(.)) = \int_0^{t_f} l(X(t), u(t))dt \tag{2}$$

is minimized.

The idea is to approach complex systems by piecewise affine models we can analytically study. Basically, an analytical approach must allow to improve approximations [14, 8]: the level of details allows to reach a compromise between quantitative quality of the approximation and the computational time.
There are many possible linearizations by parts. For instance, one can build a virtual mesh of the phase space and use multi-dimensional interpolation to define an affine approximation of the system in each cell (simplex) of the mesh, see [7],[13],[1] for more details. One can also linearize each equation separately by implicit representation and one-dimensional linearization on each variable. The latter has been done e.g. for biological systems, where simplifications in relation to real data and in regard of simulations of the model are possible, see [8].

We here choose to use a hybrid system modeling, i.e. to approximate the original system (1) by a continuous and piecewise linear one and build a virtual mesh of the phase space times the control space. The first part of this report is devoted to the resolution of any linear optimal control problem. The second one presents algorithms for the hybrid approximation of the nonlinear system.

Part I
Symbolic/Numeric Control of Affine Dynamical Systems

In this part we consider a linear dynamical system

$$\left\{ \begin{array}{lcl} \dot{X}(t) & = & AX(t) + Bu(t) \\ X(0) & = & X_0 \end{array} \right. \tag{3}$$

where $\forall t \geq 0, X(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{U}_m = \{s_1, \ldots, s_p\} \subset \mathbb{R}^m$. We want to control the system (3) from an initial state X_0 to a target $X_f = 0$ at an unspecified time t_f, in such a way that (2) is minimized.
Here, we provide a full implementation analyzing linear optimal control problems as general as possible. Our algorithm is divided in four steps:

(1) Canonical transformation (cf. §1).

(2) Computation of the controllable set (cf. §2.2).

(3) Computation of optimal solutions (cf. §3).

(4) Inverse transformation (cf. §1.2).

Each step can be done in many different ways and some salient features of our presentation are:

- a new and more efficient implementation by block matrices of the Kalman decomposition algorithm.
- a new method to compute an under-approximation of the controllable domain.

- the design of symbolic algorithms for most of the subroutines involved.
- an efficient and generic implementation of the optimal solution computation, for a very large class of cost functions.
- high dimensions treatment, even when compared to numerical softwares.

In section 1, we will develop explicit algorithms to compute the canonical transformation of any linear optimal control problem and then the inverse transformation. In section 2, we provide a controllability analysis and then in section 3, the analytical computation of optimal solutions of the canonical problem. We illustrate the algorithms and give timings in the last section.

1 Canonical Transformation

Linear control systems have been widely analyzed. In [19, 18], Kalman considers constant linear optimal control problems without constraints on the control. In this context, we have two well-known results: the first one is a controllability criterion (see [19, 6, 3] for more details), and the second is the following decomposition theorem:

Theorem 1 ([18] Kalman Canonical Structure). *Let A and B be real matrices having respective sizes $n \times n$ and $n \times m$. There exists an invertible $n \times n$ matrix T such that:*

$$T^{-1}AT = \begin{bmatrix} A_1 & A_2 \\ 0 & A_3 \end{bmatrix} \qquad T^{-1}B = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}$$

where $r = rk([B\ AB \dots A^{n-1}B]) = rk([B_1\ A_1B_1 \dots A_1^{n-1}B_1])$, A_1 is a r dimensional square matrix and B_1 a $r \times m$ matrix.

There exist many numerical algorithms computing the Kalman canonical structure of linear dynamical systems. Next, we propose a new explicit and symbolic algorithm for the Kalman decomposition. Our approach is to use block versions of the linear algebra algorithms as in [9] in order to improve the locality of the computations and treat larger problems faster.

1.1 Block Canonical decomposition

We consider the general linear system (3). Our decomposition is divided into two steps: we first reduce the system to one with a full rank mapping of the control and second apply a LQUP decomposition to the Kalman matrix.

1.1.1 Simplification to $rk(B) = m$

Lemma 1. *Let us consider the linear system (3). There exists a full rank matrix $\tilde{B} \in \mathbb{R}^{n \times rk(B)}$ and a linear mapping $\Phi \in \mathbb{R}^{rk(B) \times m}$ such that: $\dot{X}(t) = AX(t) + \tilde{B}\Phi u(t)$.*

Proof. $b = rk(B)$. If $b < m$, then there exists a column permutation $P \in \mathbb{R}^{m \times m}$ s.t.: $BP = [\tilde{B}|B_0]$ where $\tilde{B} \in \mathbb{R}^{n \times b}$ and $rk(\tilde{B}) = b$. Moreover, the column vectors of B_0 are linearly dependent of those of $\tilde{B}$, i.e.: $\exists \Lambda \in \mathbb{R}^{b \times m-b}, B_0 = \tilde{B}\Lambda$. Hence: $B = \tilde{B}\ [I_b|\Lambda]P^{-1}$ and $\Phi = [I_b|\Lambda]P^{-1}$. ☐

In the following, we will denote by $FullRank(B)$ the algorithm computing $(b, \tilde{B}, \Phi)$ from a matrix B as in the lemma.

1.1.2 Block Kalman Canonical Form

Now we want to decompose the state space of our linear system into a controllable part and an uncontrollable one. The classical method is to introduce the linear subspace $W(A, B) = span(B, AB, \ldots, A^{n-1}B)$ and then prove that W is the first subspace of $\mathbb{R}^n$ satisfying both: (i) $Im(B) \subset W$ (ii) W is A-invariant. The method is then to decompose the state space $\mathbb{R}^n$ into $W \oplus \bar{W}$: one has to compute a basis of the subspace $W(A, B)$ and to complete it for the whole state space. The matrix T of theorem 1 would be the change matrix from the canonical basis to the computed basis.

Here we propose a new approach via block matrix computation developed in collaboration with C. Pernet: we use the so-called $LQUP$ decomposition of a $x \times y$ matrix of rank r, where $U = \left[\begin{array}{c|c} U_1 & U_2 \\ \hline 0 & 0 \end{array}\right]$ is $x \times y$, U_1 is an upper triangular $r \times r$ invertible matrix, L is $x \times x$, lower block-triangular, and P and Q are permutation matrices [16].

Algorithm 1 KalmanForm

```
Require: $A$ $n \times n$ matrix, $B$ $n \times m$ matrix.
Ensure: $r, T, A_1, A_2, A_3, B_1$ as in theorem 1.
 1: $K = [B|AB|\ldots|A^{n-1}B]$;
 2: $(L, Q, U_1, U_2, P, r) = LQUP(K^T)$;
 3: if $r = n$ then
 4:     Return $(n, I_n, A, \emptyset, \emptyset, B)$.
 5: end if
 6: Form $\delta = [I_r|0]Q^T LQ \left[\begin{array}{c} I_r \\ \hline 0 \end{array}\right]$, lower triangular.
 7: Form $d = [I_{r+1..nm}|0]Q^T LQ \left[\begin{array}{c} I_r \\ \hline 0 \end{array}\right]$.
 8: $G = [I_r|0]Q^T K^T$.
 9: $C_1 = G(A^T P^T \left[\begin{array}{c} I_r \\ \hline 0 \end{array}\right] U_1^{-1}\delta^{-1})$
10: $C_2 = [0|I_{n-r}]P(A^T P^T \left[\begin{array}{c} I_r \\ \hline 0 \end{array}\right] U_1^{-1}\delta^{-1})$
11: $C_3 = [0|I_{n-r}]PA^T P^T \left[\begin{array}{c} -U_1^{-1}U_2 \\ \hline I_{n-r} \end{array}\right]$
12: $Q_1 = [I_m|0]Q \left[\begin{array}{c} I_r \\ \hline d\delta^{-1} \end{array}\right]$        {$Q_1$ is $m \times r$}
13: Return $(r, \left[\begin{array}{c} G \\ \hline [0|I_{n-r}]P \end{array}\right]^T, C_1^T, C_2^T, C_3^T, Q_1^T)$.
```

Theorem 2. *Algorithm 1 is correct and its arithmetic complexity is $O(n^\omega m)$*[1].

[1] where ω is the exponent of matrix multiplication (3 for the classical algorithm and 2.3755 for Coppersmith-Winograd's)

Proof. The full proof is given in appendix A. It has three parts and is actually another, constructive, proof of Kalman's theorem:
1. First, use the generalization of the companion matrix decomposition to prove that $GA^T = C_1 G$.
2. Second, use the latter to show that $T^{-1}AT$ is block triangular.
3. Show that $T^{-1}B$ has generic rank profile.
4. Now for the complexity: building the Kalman matrix is n matrix multiplications $n \times n$ by $n \times m$, each requiring $O(n^{\omega-1}m)$ operations. Following [10, Lemma 4.1], the LQUP decomposition requires $O\left(n^{\omega-1}(mn+n)\right)$ operations. Those two costs dominate the remaining operations: two triangular inversions $O(r^\omega)$, some permutations and column selections, and small matrix multiplications (GA^T is $O(rn^{\omega-1})$ and $d\delta^{-1}$ is $O(nmr^{\omega-1})$ where $r \leq n$.). □

Our implementation and constructive proof of the Kalman decomposition are based on LQUP factorization and block matrix computation. The better locality induced by this block version enables the use of very fast Basic Linear Algebra Subroutines, even with symbolic computations [10]. Therefore the computation time is improved. Moreover if we first apply the algorithm *FullRank* of paragraph 1.1.1, the system (3) can be replaced by another linear one:

$$\dot{Y}(t) = \begin{bmatrix} A_1 & A_2 \\ 0 & A_3 \end{bmatrix} Y(t) + \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \tilde{u}(t) \qquad (4)$$

via possibly two variable changes: $\left\{ \begin{array}{lcl} Y(t) & = & T^{-1}X(t) \\ \tilde{u}(t) & = & \Phi u(t) \end{array} \right.$
Next, we use these decomposition in order to define a canonical optimal control problem, simpler to solve.

1.2 Inverse transformation

In this section, the focus is on the explicit construction of a new linear optimal control problem under the dynamic (4). New state and control spaces, a new cost function have to be constructed and initial solutions have to be recovered.

1.2.1 Control Space

In this paragraph we focus on the construction of a new control space for our linear system (4).
By assumptions the control $u(.)$ satisfies: $\forall t \geq 0,\ u(t) \in \mathbb{U}_m = Conv(s_1, \ldots, s_p)$. Moreover the image of a polyhedron in finite dimension by a linear mapping is polyhedral. So the new control polyhedron is: $\Phi\mathbb{U}_m = Conv(\Phi s_1, \ldots, \Phi s_p)$. Note that, if $rk(B) = m$, then $\Phi = I_m$, so that no control change is needed.

When $rk(B) < m$, the main difficulty to build our optimal control problem is that there is not any invertible relation between u and $\tilde{u}$; consequently to switch from one control problem to the other, we will first need to define the pseudo-inverse of the control change matrix: $\tilde{s}_1, \ldots, \tilde{s}_{p'}$ are the vertices of ΦU_m. We introduce the Moore-Penrose pseudo-inverse $\Psi \in \mathbb{R}^{m \times b})$ [27] of the matrix $\Phi = [I_b|\Lambda]P^{-1}$:

$$\Psi = P \left[\frac{I_b}{0}\right] \text{ defined by: } \forall i \in \{1, \ldots, p'\}, \Psi\tilde{s}_i = s_k$$

where $k = min\{j \in \{1, \dots, p\} \ / \ \Phi s_j = \tilde{s}_i\}$. By linearity, Ψ is also well defined on the whole polyhedron $\Phi\mathbb{U}_m$, indeed: $\forall \tilde{u} \in \Phi U_m, \exists (\alpha_i)_{i=1\dots p'} \in [0,1]^{p'}, \sum_{i=1}^{p'} \alpha_i = 1, \tilde{u} = \sum_{i=1}^{p'} \alpha_i \tilde{s}_i$. Hence $\Psi\tilde{u} = \sum_{i=1}^{p'} \alpha_i \Psi \tilde{s}_i$ and the proposition is proven:

Proposition 1.

(i) $\Phi\Psi = I_b$

(ii) $\forall u \in \mathbb{U}_m, Bu = \tilde{B}\Phi u$

(iii) $\forall \tilde{u} \in \Phi\mathbb{U}_m, \tilde{B}\tilde{u} = B\Psi\tilde{u}$.

1.2.2 State Space

By construction, the change matrix T is non singular. Therefore, a trajectory $Y(.)$ from an initial point Y_0 corresponds to a trajectory $X(.) = TY(.)$ from the initial point $X_0 = TY_0$. Every trajectory is necessarily related to a control, the table 1 displays the correspondence between each trajectory.

Initial Problem (3)		Canonical Problem (5)
$(X(.), u(.))$	$\rightarrow$	$(T^{-1}X(.), \Phi u)$
$(TY(.), \Psi\tilde{u})$	$\leftarrow$	$(Y(.), \tilde{u}(.))$

Table 1: Corresponding trajectories

Proof. The key point here is that a trajectory $(X(.), u(.))$ in the X-space is a solution of the system (3):

$$X(t) = e^{At}X_0 + e^{At}\int_0^t e^{-Aw}Bu(w)dw$$

$$\begin{aligned} T^{-1}X(t) &= e^{(T^{-1}AT)t}T^{-1}X_0 + \ e^{(T^{-1}AT)t}\int_0^t e^{-(T^{-1}AT)w}T^{-1}Bu(w)dw] \\ &= e^{(T^{-1}AT)t}T^{-1}X_0 + \ e^{(T^{-1}AT)t}\int_0^t e^{-(T^{-1}AT)w}T^{-1}\tilde{B}\Phi u(w)dw] \end{aligned}$$

Then $(T^{-1}X(.), \Phi u(.))$ is a solution of (4), i.e. a trajectory in the Y-space. □

1.2.3 Cost Function

Let X_0 be a controllable point. The value function related to the initial control problem (3) is defined by: $V(X_0) = \inf_{u(.)}\int_0^{+\infty} l(X(t), u(t))dt$. We want to define a new value function $\tilde{V}(Y_0) = \inf_{\tilde{u}(.)}\int_0^{+\infty} \tilde{l}(Y(t), \tilde{u}(t))dt$ such that the two related optimal control problems are equivalent.

First, the idea is to define a new cost function $\tilde{l}$, such that the value function is invariant by canonical transformation (i.e.: $V(X_0) = \tilde{V}(T^{-1}X_0)$). In this case, $\tilde{l}(Y, \tilde{u}) \mapsto l(TY, \Psi\tilde{u})$ and the new optimal control problem becomes:

"Minimize $\tilde{J}(Y_0, \tilde{u}(.)) = \int_0^{+\infty} \tilde{l}(Y(t), \tilde{u}(t))dt$ with respect to the control $\tilde{u}(.)$ under the dynamic (4) and the constraints: $\forall t \geq 0, \tilde{u}(t) \in Conv\{\tilde{s}_1, \dots, \tilde{s}_{p'}\}$". (5)

We then have to verify that optimal solutions of this new problem correspond to optimal solutions of (3):

Proposition 2. *Let $(Y^*(.), \tilde{u}^*(.))$ be an optimal solution of (5).*
Then $(TY^*(.), \Psi\tilde{u}^*(.))$ is an optimal solution of the initial problem (3) and $V(TY_0) = \tilde{V}(Y_0)$.

The proof is by inspection of $J(X_0, \Psi\tilde{u}^*)$ and is given in appendix B.1.

1.2.4 Algorithms

To conclude the section, we describe two algorithms: *SimplifySystem* and *InverseTransformation*. From one given optimal control problem, *SimplifySystem* allows to define a canonical optimal control problem (see §1.1) ; once this problem is solved, *InverseTransformation* compute the related optimal solutions of the initial problem (3).

In the following algorithms, the pseudo-inverse Ψ of Φ is given e.g. by [27]. Now, if T, Ψ are transformation matrices and $(Y^*, \tilde{u}^*)$ optimal solutions of the

Algorithm 2 SimplifySystem

```
Require: $A$, $B$, $U_m = [s_1, \dots, s_p]$, $l$.
Ensure: $r, T, \Phi, \Psi, A_1, A_2, A_3, \tilde{U}, \tilde{l}$. (Data for the new optimal control problem:
    r, the state change matrix, the control change, the dynamic, the control
    space and the cost function).
    {Definition of the new control space:}
 1: $(b, \tilde{B}, M)$:=FullRank(B);
 2: $\Psi := PseudoInverse(\Phi)$;
 3: $\tilde{U}$ := ConvexHull($\Phi s_1, \dots, \Phi s_p$);
    {Definition of the new optimal control problem:}
 4: $(r, T, A_1, A_2, A_3, B_1) := KalmanForm(A, \tilde{B})$;
    {Definition of the new cost function:}
 5: $\tilde{l} := (Y, \tilde{u}) \mapsto l(TY, \Psi\tilde{u})$
 6: Return $(r, T, \Phi, \Psi, A_1, A_2, A_3, \tilde{U}, \tilde{l})$.
```

new optimal control problem (cf. section 3), then algorithm *InverseTransformation* just makes use of proposition 2.

Algorithm 3 InverseTransformation

```
Require: $T, \Psi, Y^*, \tilde{u}^*$.
 1: Return $(TY^*, \Psi\tilde{u}^*, \mathbb{U}_m, \Phi\mathbb{U}_m)$.
```

In this section we achieved the transformation of any linear optimal control problem into a canonical one. Moreover we have proved that optimal solutions of the canonical problem give optimal solutions of our initial problem. We have also proposed algorithms for switching to one problem to the other. Now, we can work on the canonical problem.

2 Controllable Domain

In this section, we consider the canonical optimal control problem previously defined and raise the question of its controllability: how to compute the set of initial points Y_0 for which the control problem (4) with the constraints $Y(0) = Y_0$; $Y(t_f) = 0$ and $\forall t \geq 0,\ u(t) \in \mathbb{U}_m = \{s_1, \dots, s_p\} \subset \mathbb{R}^m$) admits a solution.

Let us state: $\forall t \geq 0,\ Y(t) = (Y_1(t), Y_2(t))$ where: $Y_1(t) \in \mathbb{R}^r$ and $Y_2(t) \in \mathbb{R}^{n-r}$. Thus the state space splits clearly up into an uncontrollable part ($\dot{Y}_2 = A_3 Y_2$) and a controllable one ($\dot{Y}_1 = A_1 Y_1 + A_2 Y_2 + B_1 u$). We study the controllability question in the two configurations.

2.1 Stabilization of the uncontrollable part

For the uncontrollable part:

$$\dot{Y}_2(t) = A_3 Y_2(t) \tag{6}$$

Clearly, $0 \in \mathbb{R}^{n-r}$ is an equilibrium point of (6). Thus the target 0 is reachable from everywhere if 0 is a stable focus of (6). In other words, the matrix A_3 has to be stable (all its eigenvalues have negative real parts). In the following, we prove that the non-stability of A_3 involves constraints on $Y_2(0)$, so that we can easily come down to the case of a stable matrix A_3: we apply the Schur decomposition to A_3 and choose to sort its eigenvalues in such a way that: $\forall i = 1 \dots k, Re(\alpha_i) < 0$ and $\forall i = k+1 \dots (n-r), Re(\alpha_i) \geq 0$. Then there exists a unitary $Q \in \mathbb{C}^{n \times n}$ such that: $Q^* A_3 Q = D + N$ where $D = diag(\alpha_1, \dots, \alpha_{n-r})$ and $N \in \mathbb{C}^{(n-r)\times(n-r)}$ is strictly upper triangular. Moreover (6) is easily solvable: $\forall t \geq 0, Y_2(t) = e^{A_3 t} Y_2(0)$. Hence:

$$\begin{aligned} Q^* Y_2(t) &= e^{Q^* A_3 Q t} Q^* Y_2(0) \\ &= e^{(D+N)t} Q^* Y_2(0) = e^{Dt} e^{Nt} Q^* Y_2(0) \\ &= \begin{bmatrix} e^{\alpha_1 t} & \star & \star \\ & & \star \\ & & e^{\alpha_{n-r} t} \end{bmatrix} Q^* Y_2(0) \end{aligned}$$

Nevertheless we **do not need to compute** e^{Nt}. Indeed, we can recursively show (by starting from $n-r$ to $k+1$) that: $([0|I_{n-r-k}]Q^*)Y_2(0) = 0$
Hence:

$$\forall t \geq 0,\ ([0|I_{n-r-k}]\ Q^*)Y_2(t) = 0$$

So under the variable change: $\tilde{Y}_2 = (Q^* Y_2)_{1..k}$, the system (4) then becomes:

$$\left\{ \begin{array}{lclcl} \dot{Y}_1(t) & = & A_1 Y_1(t) & + & \tilde{A}_2 \tilde{Y}_2(t) + B_1 u(t) \\ \dot{\tilde{Y}}_2(t) & = & & & \tilde{A}_3 \tilde{Y}_2(t) \end{array} \right.$$

where: $\tilde{A}_2 = A_2 Q \left[\dfrac{I_k}{0}\right]$ and $\tilde{A}_3 = (D+N)\left[\dfrac{I_k}{0}\right]$ is stable.

We have shown that the analysis of the uncontrollable part of the system (4) leads to define a subspace of the state space, namely $\{(Y_1, \tilde{Y}_2, 0) \in \mathbb{R}^r \times \mathbb{R}^k \times \mathbb{R}^{n-r-k}\}$. In this subspace, $\tilde{Y}_2(.)$ trajectories converge towards 0. From now on, we therefore restrict our analysis to a system (4) where the matrix A_3 is stable.

2.2 Under-Approximation of the Controllable Domain

Now, we assume w.l.o.g that the points s_i defining the control boundaries are such that: $s_i \notin Conv_{j \neq i}(s_j)$. Therefore, each point s_i is a vertex of the polytope

$\mathbb{U}_m$ and we have (cf §1): $rk(B) = m$, $rk([B|AB|\dots|A^{n-1}B]) = n$. We want to find the set of controllable points of our system. By time reversal, we come down to the computation of the attainable set from the target point 0. In [1], for safety verification, the idea is to compute a conservative over-approximation of the attainable set. They can thus certify that the system can not escape from an admissible set of states. On the contrary, we need a guaranty that Y_0 is controllable. Therefore we instead compute an under-approximation of this set.

Let us start by defining the controllable set C in our context: $C = \{Y \in \mathbb{R}^n / \exists T \geq 0, \exists u : [0,T] \to U_m, Y = \int_0^T e^{-A\tau}Bu(\tau)d\tau\}$. Indeed, any solution of a linear system $\dot{Y}(t) = AY(t) + Bu(t)$ has the form: $Y(t) = e^{At}Y(0) + \int_0^t e^{A(t-s)}Bu(s)ds$.

Proposition 3. *The controllable domain C is a convex subset of the state space.*

The proof is given in appendix B.2. It defines (by convexity and at maximal time) a new control from that of some controllable points within C.

Now we can introduce our under-approximation of the domain by time-reversal of the control polytope:

Corollary 1. *Let $Y_i(.)$ be the trajectory from 0 by time reversal according to $u = s_i$. If $C(t) = Conv_{1..k}(Y_i(t))$, then*

$$C(t) \subset C \text{ and } \forall Y \in C(t), \exists \text{ a control } u, Y = \int_0^t e^{-A\tau}Bu(\tau)d\tau.$$

Any point in $C(t)$ is said controllable at least in time t *and $C(t)$ is an* under-approximation *of the controllable set in time t.*

This gives us an algorithm to build our under-approximation in time T. Nevertheless for a given time T, the quality of the approximation could be very poor (see example 1, figure 1-(a)). To refine it, we choose to discretize the time interval $[0,T]$ in N subintervals. The under-approximation in time T is the convex hull of under-approximations in time $j * h$ for $j = 1..N-1$ (where $h = T/N$) and the quality is significantly improved (see example 1, figure 1-(b)). We have thus defined the following algorithm, *UnderApproximation*, computing a set of controllable points.

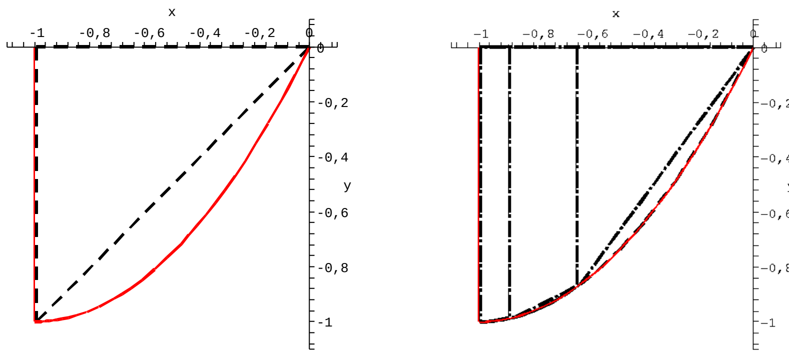
Algorithm 4 UnderApproximation

```
Require: $A, B, U, T, h$ (where $U = Conv\{s_1, \dots, s_p\}$).
Ensure: an under-approximation with a step $h = T/N$ of the controllable
    domain in time T.
1: ApproxVertices:=[0];
2: for all time step j (from 1 to N) do
3:   for all vertex $s_i$ do
4:     $Y_i(.)$ = trajectory from 0 with $u = s_i$;
5:     ApproxVertices:=ApproxVertices $\cup\{Y_i(jh)\}$;
6:   end for
7: end for
8: Return ConvexHull(ApproxVertices);
```

Example 1 (2D Under-Approximations). *Let us consider the system:*

$$\dot{Y} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} Y + \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix} u$$

with $u \in Conv(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix})$. *The following figures show in dashes under-approximations of the controllable set (represented in plain line) for three refinements.*



(a) (b)

Figure 1: Under-approximations in time $T = 5$ (a) without refinement ($N = 1$) (b) by discretizing ($N = 5$ in dash-dots - $N = 30$ in dashes)

3 Optimal Solutions

In this section, we present some theoretical results and algorithms for solving linear optimal control problems. The algorithm is as general and symbolic as possible to approximate optimal controllers. Recall that we want to control a linear system:

$$\dot{Y}(t) = AY(t) + Bu(t), \qquad A \in \mathbb{R}^{n\times n}, \qquad B \in \mathbb{R}^{n\times m}$$

from a controllable initial state Y_0 to a final state $Y_f = 0$ at an unspecified time t_f using the admissible control functions $u \in \mathbb{U}_m = Conv_{1..p}(s_p) \subset \mathbb{R}^m$ in such a way that: $J(Y_0, u(.)) = \int_0^{+\infty} l(Y(t), u(t))dt$ is minimized. According to the decomposition algorithm developed in section 1, we also assume: $rk(B) = p$ and $rk([B|AB|\dots|A^{n-1}B]) = n$. To solve this canonical system, we now introduce the Hamiltonian function:

$$H(Y, u, \lambda) = l(Y, u) + \lambda^T AY + \lambda^T Bu$$

The pseudo-Hamiltonian formulation of the optimal control problem and the Pontryagin Minimum principle provide us the following optimization problem [24, §1], [5, §2],[23, §4]:

$\mathcal{P}$: "Minimize H with respect to the control variable $u \in \mathbb{U}_m$ under the constraints:

$$\dot{Y}(t) = \frac{\partial H}{\partial \lambda}(Y(t), u(t), \lambda(t)) \tag{7}$$

$$\dot{\lambda}(t)^T = -\frac{\partial H}{\partial Y}(Y(t), u(t), \lambda(t)) \tag{8}$$

and $H(Y^\star(t), \lambda^\star(t), u^\star(t)) = 0$ along the optimal trajectory."

Our algorithm is divided in two main steps: first, the controllable set is partitioned (cf 3.1) in domains $(\tilde{\Gamma}_i)_{i \in I}$, inside which the optimal control is constant (equal to s_i). In practice, we need an algorithm computing the boundaries of these cells (see §3.2). The second step requires to compute an optimal trajectory from an entry point to the target within each cell. In this section, to the cost function l is assumed linear in the control: $l(Y, u) = l_0(Y) + l_1(Y)u$. The case were this function is nonlinear is actually simple: in this case, the Hamiltonian optimization problem could be solved by classical tools. Indeed, $\frac{\partial H}{\partial u} = 0$ is then solvable in the control variable u.

3.1 Singular control

Let us consider the optimization problem $\mathcal{P}$. By definition, $\mathcal{P}$ is a linear program. It thus admits solutions which may occur on the boundary of the polyhedral set $\mathbb{U}_m$. Now, any solution (Y, u, λ) of the Hamiltonian system (7) is said to be *extremal* and distinguish regular and singular solutions:

Definition 1. *An extremal* $(Y(t), u(t), \lambda(t))$ *is called* REGULAR *on an interval* $[t_0, t_1]$*, if there exists* k *s.t., for almost all* $t \in [t_0, t_1]$, $\lambda^T(t)Bs_k < \min\{\lambda^T(t)Bs_i; i \neq k\}$

Therefore, for any regular extremal $(Y(t), u(t), \lambda(t))$, the optimal control is given by the relation:

$$u(t) = s_i \ \text{ if } \ \lambda^T(t)Bs_i < \min_{j \neq i}\{\lambda^T(t)Bs_j\}$$

Consequently the controllable domain can be partitioned in domains defined as follows:

Definition 2. *An optimal trajectory* $Y(.)$ *belongs to the* DOMAIN Γ_i *on a time interval* $[t_0, t_1]$ *if the condition:* $\forall t \in [t_0, t_1]$, $\forall j \in \{1, ..., m\} - \{i\}$, $\lambda^T(t)B(s_i - s_j) < 0$ *holds. Thus at any point of the domain* Γ_i*, the optimal control is* $u(.) = s_i$ *and the associated field vector is* $AY + Bs_i$.

Moreover if, for a regular trajectory, there exists $(i, j) \in \{1, \ldots, m\}^2$ such that: $\forall k \neq i, j$, $\lambda^T(t_1)Bs_i = \lambda^T(t_1)Bs_j < \lambda^T(t_1)Bs_k$. Then t_1 is a switching point of the control between s_i and s_j. The switching function is then:

Definition 3 (Switching function).

$$\begin{aligned} S_{i,j}(t) &= \tfrac{\partial H}{\partial u}(Y(t), u(t), \lambda(t)) \\ &= l_1(Y(t)) + \lambda^T(t)B(s_i - s_j) \end{aligned}$$

Now, the single zeros of $S_{i,j}$ give us the switching time between the domains Γ_i and Γ_j. However it may also be possible to find time intervals where the switching function is identically equal to zero. This typically corresponds to the appearance of singular arcs in each face of the polyhedral control set. Thus singular trajectories are:

Definition 4. *[28] A trajectory $Y(t)$ is called ij-SINGULAR on a time interval $[t_0, t_1]$ if the condition: $\lambda^T\ Bs_i \equiv \lambda^T\ Bs_j < \lambda^T\ Bs_k$, $k \neq, i, j$ holds for almost all $t \in [t_0, t_1]$.*

Just note that definition could be naturally extended to I-singular trajectories ($I \subset \{1, \ldots, p\}$). According to definitions 2 and 4, we show that ij-singular trajectories geometrically correspond to the boundary between Γ_i and Γ_j. On this singular boundary, the optimal control is said to be singular and we have the following result on singular controls:

Proposition 4. *Let us consider an ij-singular trajectory $Y(.)$ on a time interval $[t_0, t_1]$. Then:*

$$\forall t \in [t_0, t_1],\ u(t) \in [s_i, s_j].$$

Likewise, on an I-singular trajectory, $u(t) \in Conv_{k\in I}(s_k)$.

3.2 Boundaries computation

At this point of our analysis, we have partitioned our state space in domains delimited by:
- singular boundaries (see e.g. [23, fully optimal problem]).
- mixed and non singular boundaries (see [17, ex. 1]).
- non singular boundaries (see [24, time-optimal problems]).
We now show how to compute those boundaries.

3.2.1 Switch rules

In this paragraph we briefly describe a method to compute the allowable "switching directions" [17] in the state space.
Let us compute the switching rules between the controls s_i and s_j. In our linear control problem, the Hamiltonian has the form: $H(Y, u, \lambda) = H_0(Y, \lambda) + H_1(Y, \lambda)u$. From examination of the sign of $\frac{d}{dt}H_1(Y(t), \lambda(t))$ at switching points (i.e. $H_1(Y(t), \lambda(t)) = 0$ and $H_0(Y(t), \lambda(t)) = 0$), it is possible to determine whether switchings from $u = s_i$ to $u = s_j$ are allowed in a given region of the state space.

3.2.2 Singular boundaries

In this paragraph we present an algorithm for computing singular boundaries when they exist. This algorithm is essentially based on the Pontryagin maximum principle [24] and classical results in the theory of singular extrema (see [20, 25, 5] for more details). It uses H with the form: $H(Y, v, \lambda) = l(Y, s_j + (s_i - s_j)v) + \lambda^T AY + \lambda^T B(s_j + (s_i - s_j)v)$ where $v \in [0, 1]$ (indeed $u \in [s_i, s_j]$ with proposition 4).

We show on table 2 some performances of this algorithm in high dimension where U_m is a random simplex in $\mathbb{R}^m$ and $n = m$:

Algorithm 5 ij-singular boundary

```
Require: $i$ and $j$, indices of the considered $\Gamma$ domains.
Require: $H(Y, v, \lambda)$.
Ensure: $\varphi$, where $\varphi(X) = 0$ defines the $ij$-boundary
Ensure: $u^\star$ the $ij$-singular optimal control.
Ensure: $\lambda^\star$ the optimal Pontryagin parameter.
1: $H_v = \frac{\partial H}{\partial v}$.
2: Compute the smallest integer $K$ such that: $\frac{\partial}{\partial v}(\frac{d^{2K}}{dt^{2K}} H_v) \neq 0$.
3: if The Legendre-Clebsh (LC) condition [20, 25]: $(-1)^K \frac{\partial}{\partial v}(\frac{d^{2K}}{dt^{2K}} H_v) \geq 0$ is not satisfied then
4:     Return "no singular solution".
5: end if
6: Solve the system $\{H = 0, H_v = 0, \left(\frac{d^i}{dt^i} H_v = 0\right)_{i=1..2K}\}$ {we then obtain not only the singular values of $v$ and $\lambda$ in relation with $Y$ but also the equation ($\varphi(Y) = 0$) of the boundary.}
7: Return $(\varphi(Y), s_j + (s_i - s_j)v(Y), \lambda(Y))$.
```

n	2	3	4	5	6	7	8
cpu (s)	0.16	0.22	0.35	0.56	0.91	1.51	2.43
n	**9**	**10**	**11**	**12**	**13**	**14**	**15**
cpu (s)	4.21	7.03	10.53	19.06	31.38	53.85	94.18

Table 2: Singular boundaries timings

Note that we still have to check that the so-computed boundary really exist in the controllable domain and that the switching conditions are satisfied: $\forall k \notin \{i, j\}, S_{i,k} < 0$. However, we show next that these conditions are not always sufficient to determine if a computed singular boundary is valid or not. Such cases appear when the computed singular control explicitly depends of the state Y. While the related boundary is bounded, the whole boundary between Γ_i and Γ_j is necessarily also made of a regular part. The next paragraph is devoted to its computation.

3.2.3 Mixed boundaries

In this paragraph we assume that we have already computed the singular boundary between two domains Γ_i and Γ_j and check the existence condition of these boundary. So we have its equation: $\varphi(Y) = 0$ under the constraint $0 \leq v(Y) \leq 1$, the singular control u^* and the related λ^*. We now want to compute the related regular boundary:

In the implementation of the *MixedBoundary* algorithm, the last step is algorithmically quite hard to achieve. Our solution is to discretize the given singular boundary, so that step 3. computes a discretization of the regular part of the whole boundary. The following example shows that this algorithm enables new resolutions.

Algorithm 6 MixedBoundary

```
Require: $i$ and $j$, indices of the considered $\Gamma$ domains.
Require: $\varphi$ equation of the $ij$-boundary.
Require: $\lambda^\star$ optimal Pontryagin parameter on the $ij$-boundary.
Ensure: a parameterization of the non singular boundary between $\Gamma_i$ and $\Gamma_j$.
1: Parameterize the singular boundary (by the implicit functions theorem)
   :$\psi(\xi)$ (i.e. such that $\varphi(\psi(\xi)) = 0$).
2: Compute the trajectory $Y[\psi(\xi), s]$ from $\psi(\xi)$ by time reversal with $u = s$
   and the corresponding $\lambda[\psi(\xi), s]$ via the Euler-Lagrange equations (8) with
   the initial condition $\lambda[\psi(\xi), s](0) = \lambda^*$.
   $s \in \{s_i, s_j\}$ is chosen according the switching rules (cf §3.2.1 i.e., e.g.,
   $Y[\psi(\xi), s_i](.)$ has to evolve in the domain ("Allowable switch from $u = s_j$
   to $u = s_i$").
3: Compute the first time $t(\xi) < 0$ for which the switching condition be-
   tween $s_i$ and $s_j$ holds (cf definition 3). The switching curve is defined
   by: $Y[\psi(\xi), s_i](t(\xi)) = 0$.
   No solution $t(\xi)$ invalidates the singular boundary so that the boundary
   between $\Gamma_i$ and $\Gamma_j$ is necessarily regular.
```

Consider the system [17, Example 1]:

$$\left\{ \begin{array}{rcl} \dot{X}(t) & = & \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} X(t) + \begin{bmatrix} 1 \\ -1 \end{bmatrix} u(t) \\ X(0) & = & X_0 \end{array} \right. \tag{9}$$

where $X \in \mathbb{R}^2$, $|u| < 1$ and the performance index to be minimized is: $J(X, u(.)) = \int_0^T \frac{1}{2} x_1(t)^2 dt$. (Note that (9) already is under its canonical form since $rk([B|AB]) = 2$). [17] provides the full analysis of the singular controls and the graph of the allowable switching regions [17, fig. 5]. There, the singular boundary is defined by $x_1 = -2x_2$ and $-1 \leq x_2 \leq 1$ and the switching function is $S(t) = 2(\lambda_1(t) - \lambda_2(t))$. Due to the constraint $-1 \leq x_2 \leq 1$, the singular boundary does not allow us to draw a partition of the state space. However, we are able to complete [17]'s results by the computation of the whole boundary between the controls $u = 1$ and $u = -1$: as we have a valid singular boundary, we can now apply our *MixedBoundary* algorithm as shown on figure 2:
1. Discretize the singular boundary $[A, B]$ in k points (x_i).
2/3. For each discretization point x_i: compute the trajectory $X[x_i, 1]$ from x_i according to $u = 1$ by time reversal ; compute the related λ with: $S(0) = 2(\lambda_1(0) - \lambda_2(0)) = 0$ via Euler-Lagrange equations ; and compute the first time T_i such that: $S(T) = 0$.
Repeat this step with $u = -1$.
4. $\{X[x_i, 1](T_i); i = 0 \ldots (k-1)\}$ is the searched discrete approximation of the boundary.

3.2.4 Non Singular Boundaries

In this paragraph, we consider the case where there is no singular or mixed boundary between the two domains Γ_i and Γ_j. The optimal control is then called **bang-bang** (i.e. piecewise constant with values in $\{s_i, s_j\}$). Let us distinguish two possible configurations:

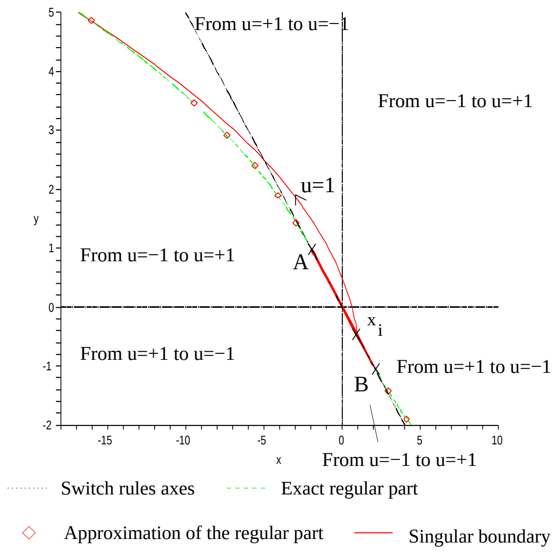


Figure 2: MixedBoundary in Gibson's problem

- For all $(i, j) \in \{1, \dots, m\}^2$, $i \neq j$, the boundary between Γ_i and Γ_j is non singular.
In this case, one method is to compute all the switching functions $S_{i,j}$ (cf definition 3). After that we can study the zeros of $S_{i,j}$ and deduce the transition time τ between two of them. By time reversal, we start from the origin and build the switching curve. This method is well described in [23].
- There exists $k \in \{1, \dots, m\} - \{i, j\}$, such that the boundary between Γ_j and Γ_k exists and is singular. In this case we come down to the same technique than for mixed boundaries: the idea is to take a parameterization of the singular boundary between Γ_j and Γ_k. We consider by time reversal the trajectory from a point of this boundary according to $u = s \in \{s_j, s_k\}$ and compute the first negative time for which the switching condition: $S_{j,k}(t) = 0$ holds.

This latter algorithm, $NonSingBoundary$ is based on the following proposition:

Proposition 5. *Let $Y(.)$ be an optimal trajectory from an initial point Y_0 and $u(.)$ the associated optimal control. We assume that there exists a time $T > 0$, such that: $\exists \epsilon > 0$, $Y(.)$ regular over $[T - \epsilon, T]$ and $Y(T + .)$ is ij-singular. Then: $\forall t \in [T - \epsilon, T]$, $u(t) \in \{s_i, s_j\}$.*

In conclusion, we have proposed three algorithms to compute boundaries between the domains Γ_i. We can now define a general one *Boundary* that compute the boundary between two given domains Γ_i and Γ_j: $(\varphi, \omega, u) := Boundary(i, j)$ where φ is the equation of the boundary, ω is defined by: $\omega(Y) = \begin{cases} 1 \text{ if } \varphi(Y) = 0 \text{ is singular} \\ 0 \text{ otherwise} \end{cases}$ and u^* the related optimal control. We therefore have all the necessary subroutines to solve our problem.

3.3 Linear Optimal Control

In this section, we detail the general algorithm for solving any linear optimal control problem. The principle is as follows: after a virtual partition of the state space in Γ_i domains, one follows the trajectory within each cell of the partition. Indeed, in each cell and in every boundaries, the control is known thanks to the algorithms of section 3.2. When the trajectory reaches a boundary, there is a switching of cell and a control change. This goes on till the target is reached.

Algorithm 7 LinearOptimalControl

```
Require: $A$, $B$, $Y_0$, $l(Y,u) = l_0(Y) + l_1(Y)u$ and $\{s_1, \dots, s_m\}$.
Ensure: Optimal trajectory, control and value function.
 1: $V := 0$; {Initialize switching functions}
 2: $S_{i,j} = \frac{\partial}{\partial u} H(Y, s_j + (s_i - s_j)v, \lambda) = l_1(Y) + \lambda^T B(s_i - s_j)$. {Virtual partition
    of the state space}
 3: $I = \{i \in [|1, m|]/\{\lambda/\forall j \neq i, S_{i,j} < 0\} \neq \emptyset\}$.
 4: for all $i \in I$ and $j \in I$ such that $i < j$ do
 5:    $(\varphi_{i,j}, \omega_{i,j}, u_{i,j}) := Boundary(i, j)$.
 6:    $(\tilde{\Gamma}_j)_{j \in I}$ is the induced partition of the controllable set.
 7: end for
    {Identification of the domains where $u = s_i$}
 8: for all $j \in I$ do
 9:    if $\partial \tilde{D}_j == \bigcup_k \{Y$ s.t. $\varphi_{i,k}(Y) = 0\}$ then
10:       $\Gamma_i := \tilde{\Gamma}_j$.
11:    end if
12: end for
13: $k := 0$; $T_0 := 0$;
    {Within each cell, reach the boundary}
14: while $Y_k \neq 0$ do
15:    Find i s.t. $Y_k \in \overline{\Gamma_i}$.
16:    if $Y_k \in \partial \Gamma_i$ then
17:       Find $j$ s.t. $\varphi_{i,j}(Y_k) = 0$.
18:       $u := u_{i,j}(Y_k)$;
19:       if $\omega_{i,j}(Y_k) == 1$ {0 is reached on this $ij$-singular boundary.} then
20:          $T_{k+1} :=$ Solution of $Y[Y_k, u](t) = 0$;
21:          break while loop;
22:       end if
23:    else
24:       $u := s_i$;
25:    end if
       {Piecewise solution}
26:    Compute $T_{k+1} = \inf\{t > 0; Y[Y_k, u](t) \in \partial \Gamma_i\}$
27:    $Y_{k+1} := Y[Y_k, u](T_{k+1})$.
28:    $u^\star := u$ for $t \in [T_k, T_{k+1}]$;
29:    $Y := Y[Y_k, u]$ for $t \in [T_k, T_{k+1}]$;
30:    $V := V + \int_0^{T_{k+1} - T_k} l(Y[Y_k, u](t)dt$
31: end while
32: Return $(Y, u^\star, V)$
```

Once the canonical problem is solved with algorithm 7, we just have to apply the inverse transformation 3 to come down to optimal solutions of our initial control problem.

3.4 Implementation in high dimension

In general, examples are developed in dimension $n = 2$ or $n = 3$ for obvious reasons of display. In usual applications, optimal control problems are solved until the dimension 5. With the expansion of the aerospace, today algorithms have to deal with dimension 6 and 7.All the algorithms presented in this paper have been implemented in Maple and work in high dimensions[2].

Part II
Hybridization of Nonlinear Dynamical Systems

We now consider the nonlinear optimal control problem (1)-(2). In the next, we present a hybrid algorithm controlling the system (1) from an initial state X_0 at time $t = 0$ to a final state $X_f = 0$ at an unspecified time t_f.
Our approach is to approximate the nonlinear dynamic (1) by a continuous and piecewise affine system. Then in each cell of the implicit build automaton, the system is fully affine ($\dot{X}(t) = AX(t) + Bu(t) + c$), so that algorithms of part I are available. Our algorithm is divided in two main steps:

(1) Hybridization of the system (1) in §4

(2) Solving the so build hybrid optimal control problem (see §5)

  (a) Under-approximation of the hybrid controllable set (§5.1)

  (b) Hybrid solving in (§5.4)

In section 4, we describe the algorithm fot the hybrid approximation of nonlinear optimal control problems. Then, in section 5, we propose a method to solve the so-build hybrid optimal control problems: we first develop a new algorithm to under-approximate the hybrid controllable set in §5.1 and then, describe a method to compute hybrid local optimal solutions.

4 Hybrid approximation of nonlinear systems

Let us consider the system:

$$\dot{X}(t) = f(X(t), u(t))$$

where $X(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{U}_m$. The polytope $\mathbb{U}_m$ is defined as the convex hull of a finite number of points in $\mathbb{R}^m$: $\mathbb{U}_m = Conv(\mathrm{s}_1, \dots, \mathrm{s}_\mathrm{k})$, such that: $0 \in \mathbb{U}_m$.

[2]The maplets are available online at: `www-lmc.imag.fr/lmc-mosaic/Jean-Guillaume.Dumas/SHOC`

Moreover we assume that the points s_i do not belong to the convex hull of the rest points $s_j,\ j \neq i$, so that each point s_i is a vertex of the polytope $\mathbb{U}_m$.

In this section the nonlinear optimal control problem (1)-(2) is replaced by a hybrid one: first we present an algorithm to approach the nonlinear dynamic (1) by a continuous and piecewise affine system (see §4.1). Then this approximation allows us, in §4.3, to define a hybrid control problem.

4.1 Hybridization

Let $h > 0$ be the discretization step. We want to approximate the differential inclusion (1) by: $\dot{X}(t) = f_h(X(t), u(t))$ where f_h is a piecewise affine approximation of the vector field f.

Let us consider the field vector f as a function of $\mathbb{R}^{n+m}$. We build $(\Delta_i)_{i \in I}$, a mesh of the space $\mathbb{R}^n \times \mathbb{U}_m$, where Δ_i is a simplex. As is done e.g. in [13], we want to approximate f by interpolation at the vertices of Δ_i. An affine function in dimension n+m is uniquely defined by its values in (n+m+1) affine independent points. Therefore, let us consider the vertices $Z_1, \ldots Z_{n+m+1}$ of the simplex Δ_i. Δ_i is then the convex hull of the points $Z_1, \ldots, Z_{n+m+1}$.

Now we can compute the approximation $f_i(X, u) = A_i X + B_i u + c_i$ of f by interpolation at the vertices of Δ_i:

$$\forall j \in \{1, \ldots, n+m+1\},\ f(Z_j) = \left[\ A_i \mid B_i\ \right] Z_j + c_i$$

i.e., for all $j = 1, \ldots, n+m+1$:

$$f(Z_j) - f(Z_1) = \left[\ A_i \mid B_i\ \right] (Z_j - Z_1)$$

We define M_i the $(n+m) \times (n+m)$ matrix, whose columns are the vectors: $\{Z_j - Z_1; j = 2, \ldots, n+m+1\}$ and F_i the $(n+m) \times n$ matrix, whose columns are the vectors: $\{f(Z_j) - f(Z_1); j = 2, \ldots, n+m+1\}$). By linear independence of the vertices of the simplex Δ_i, M_i is non singular. So we obtain:

$$\begin{aligned} \left[\ A_i \mid B_i\ \right] &= F_i (M_i)^{-1} \\ c_i &= f(Z_1) - \left[\ A_i \mid B_i\ \right] Z_1 \end{aligned} \tag{10}$$

Remark 1. *The c_i can be solved as well with any column:*

$$\forall j = 1, \ldots, n+m+1,\ f(Z_j) - \left[\ A_i \mid B_i\ \right] Z_j = c_i$$

Consequently the piecewise affine approximation of (1) is defined by:

$$f_h(X(t), u(t)) = A_i X(t) + B_i u(t) + c_i,\ \text{if } (X(t), u(t)) \in \Delta_i$$

The key point is how to define the mesh. We show now that a meshing of the whole space is not mandatory. We rather compute the mesh as we go.

4.2 Implicit simplicial mesh

In previous section, we have seen how to build a piecewise affine approximation of the nonlinear dynamic f for a given simplicial mesh of the state and control domain. So to perform our hybrid approximation, we need now a method to build a simplicial mesh of $\mathbb{R}^n \times \mathbb{U}_m$.
As $\mathbb{R}^n \times U_m$ is not bounded, it's algorithmically inconceivable to mesh the whole space $\mathbb{R}^n \times \mathbb{U}_m$. Our approach is then to implicitly define a mesh of our space, so that the simplicial subdivision is made on the fly (see section 5).

Let $h > 0$ be the discretization step introduced in §4.1. Our algorithm is divided into three main parts: first we build a simplicial mesh $(D_i)_i$ of the state space $\mathbb{R}^n$. Then we compute a triangulation of the control domain $\mathbb{U}_m$. Lastly we deduce a simplicial mesh $(\Delta_i)_{i\in I}$ of $\mathbb{R}^n \times \mathbb{U}_m$. Moreover we impose:

(P_1) If $0 \in \Delta_i$, then 0 is a vertex of Δ_i

(P_2) The mesh $(D_i)_i$ is the projection of $(\Delta_j)_j$ over $\mathbb{R}^n$

The property (P_1) ensures that 0 is an equilibrium point of the piecewise affine approximation f_h (Indeed if Δ_i is a cell of our so computed mesh and $\dot{X} = A_iX + B_iu + c_i$ the related dynamic, then, according to remark 1, we have: $0 \in \Delta_i \Rightarrow c_i = 0$). The property (P_2) will allow us in section 5 to have the same output constraints in each cell Δ_i such that: $\forall i, p_{\mathbb{R}^n}(\Delta_i) = D_q$.

Simplicial mesh of $\mathbb{R}^n$ The state space $\mathbb{R}^n$ is implicitly cut into n-dimensional cubes. Each cube is then meshed into $n!$ simplices and the resulting partition is a mesh of $\mathbb{R}^n$ (see [12],[14, chapter 11]) for more details):
Let us consider a n-cube $C = [a_1, a_1 + h] \times \cdots \times [a_n, a_n + h]$. We then introduce S_n the set of permutations of $\{1, \ldots, n\}$. $D_\varphi = \{(x_1, \ldots, x_n) \in \mathbb{R}^n; 0 \leq x_{\varphi(1)} - a_{\varphi(1)} \leq \cdots \leq x_{\varphi(n)} - a_{\varphi(n)} \leq h\}$ is a simplex in $\mathbb{R}^n$, whose vertices are defined by:

$$\left\{ \begin{array}{ll} \forall i = 1, \ldots, p, & x_{\varphi(i)} = a_{\varphi(i)} \\ \forall i = p+1, \ldots, n, & x_{\varphi(i)} = a_{\varphi(i)} + h \end{array} \right. , p = 0, \ldots, n \tag{11}$$

$(D_\varphi)_{\varphi\in S_n}$ is [12] a mesh of the cube C.

Triangulation of $\mathbb{U}_\mathbf{m}$ $\mathbb{U}_m$ is a bounded polyhedron, defined as the convex hull of its vertices. The study of simplicial subdivisions of such polytopes has been extensively developed in recent years and provides us some efficient tools to compute them (e.g. Delaunay triangulation of the polymake software[3]).

Simplicial mesh of $\mathbb{R}^n \times \mathbb{U}_\mathbf{m}$ We let $(D_i)_i$ and $(U_j)_j$ respectively be the simplicial subdivisions of $\mathbb{R}^n$ and $\mathbb{U}_m$. We just build the triangulation $(\Delta_q)_q$ of $\mathbb{R}^n \times \mathbb{U}_m$ **without new vertices**. This last criterion guaranties the property (P_2) to be satisfied.

In this section, we have described the main steps of the construction of an implicit simplicial mesh. In the next, we will see that this mesh actually defines a hybrid approximation of the initial dynamical system (1).

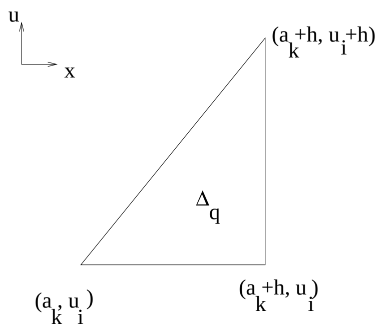
[3]tool for the algorithmic treatment of convex polyhedra and finite simplicial complexes by E. Gawrilow and M. Joswig, http://www.math.tu-berlin.de/diskregeom/

4.3 Hybrid automaton

In sections 4.1 and 4.2, we build a piecewise affine approximation of the nonlinear vector field f over a mesh $\Delta = (\Delta_q)_{q \in I}$ of the space $\mathbb{R}^n \times \mathbb{U}_m$.

We thus have defined a hybrid automaton $\mathcal{H} = (\mathcal{Q}, \mathcal{D}, \mathcal{U}, \mathcal{E}, \mathcal{F}, \mathcal{G})$ (similar to those of [7, 14]), as follows:

1. $\mathcal{Q}$ the countable set of the indices of the simplexes Δ_q.
2. $\mathcal{D} = \{D_q \ / \ q \in \mathcal{Q}\}$ the collection of domains induced by Δ over the state space: $\forall q \in \mathcal{Q}, D_q = p_{\mathbb{R}^n}(\Delta_q)$
$$\forall (q, q') \in \mathcal{Q}^2, [int(D_q) \cap int(D_{q'}) \neq \emptyset \Rightarrow D_q = D_{q'}]$$
3. $\mathcal{U} = \{U_q \ / \ q \in \mathcal{Q}\} \subset \mathbb{U}_m$ the collection of control domains induced by Δ (note that U_q depends on the state X, see figure 3).
4. $\mathcal{E} = \{(q, q') \in \mathcal{Q} \times \mathcal{Q} / \ \partial D_q \cap \partial D_{q'} \neq \emptyset\}$ the transition set.
5. $\mathcal{F} = \{f_q \ / \ q \in \mathcal{Q}\}$ the collection of the affine field vectors of section 4.1:
$$\boxed{\begin{array}{llll} f_q : & \Delta_q & \rightarrow & \mathbb{R}^n \\ & (X, u) & \rightarrow & A_q X + B_q u + c_q \end{array}}$$
6. $\mathcal{G} = \{G_e \ / \ e \in \mathcal{E}\}$ the collection of the guards: $\forall e = (q, q') \in \mathcal{E}, G_e = \partial D_q \cap \partial D_{q'}$
7. $\mathcal{R} = \{R_e / e \in \mathcal{E}\}$ the collection of Reset functions: $\forall e = (q, q') \in \mathcal{E}, \forall x \in G_e, R_e(x) = \{x\}$ (Here, we do not need to reinitialize the continuous variable x, since the D_q are adjacent).



$$\begin{gathered} (x, u) \in \Delta_q \\ \Updownarrow \\ x \in D_q = [a_k, a_k + h] \\ \text{and} \\ u \in U_q = \{u \in \mathbb{U}_1; 0 \leq u - u_i \leq x - a_k\} \end{gathered}$$

Figure 3: Definition of D_q and U_q in cell Δ_q for $n = m = 1$

By definition, D_q is the set of state constraints and U_q the set of mixed state and control constraints in the mode q. Moreover the Δ_q are simplices in $\mathbb{R}^{n+m}$, so that these (inequality) constraints are both linear in the state and in the control. From now on, we will make the following assumption:

Hypothesis 1. *The hybrid automaton $\mathcal{H}$ is assumed not Zeno*[4]

[4]Zeno executions correspond to an infinite number of switch in a finite time. That often involves problems in the simulation of hybrid system. Indeed the transition times come closer and closer and in simulations, we can not differentiate them any more (see [14, 29]).

Once the hybrid automaton $\mathcal{H}$ is defined, we can then build a new optimal control problem $(\mathcal{P}_{\mathcal{H}})$ as the hybrid approximation of the initial one (1)-(2): Let X_0 be a point of $\mathbb{R}^n$. We want to find a trajectory under the $\mathcal{H}$'s dynamic that steers the initial point X_0 to the target point 0, locally minimizing the cost function $J(X_0, u(.))$.

In the next paragraph, we will define the notion of solution of the hybrid optimal control problem. We will also present an algorithm for solving this problem.

5 Solving the hybrid optimal control Problem

In §4, we have defined a hybrid automaton H approaching the behavior of our initial problem (1)-(2). In this section we present a method for solving the related hybrid optimal control problem $(\mathcal{P}_{\mathcal{H}})$.

Let us start by defining the solutions of the hybrid control problem: to guarantee existence and uniqueness of solutions of the system (1), the function f has to satisfy the assumptions of the Cauchy-Lipschitz theorem. In addition we assume the function l to have the same regularity properties than f:
- $f : \mathbb{R}^n \times \mathbb{U}_m \to \mathbb{R}^n$ continuous application, Lipschitz in the variable X and bounded on $\mathbb{R}^n \times \mathbb{U}_p$. Furthermore: $f(0,0) = 0$.
- $l : \mathbb{R}^n \times \mathbb{U}_m \to \mathbb{R}^n$ -instantaneous cost- continuous (non-coarse) application, Lipschitz in x and bounded on $\mathbb{R}^n \times \mathbb{U}_p$. Furthermore: $l(0,0) = 0$.

Under these assumptions, the system (1) admits an unique solution $X[u, X_0]$.

The optimal control problem consists in computing the best admissible control u that steers the state from X_0 to the target point 0 at an unspecified final time.

5.1 Hybrid automaton controllability

In this paragraph, we want to compute the set of controllable points in $\mathbb{R}^n$, i.e. the set of initial points for which the hybrid problem $(\mathcal{P}_{\mathcal{H}})$ admits a solution. By time reversal, we come down the computation of the attainable set from 0. In [1], for safety verification, the idea is to compute a conservative over-approximation of the attainable set. They can thus certify that the system can not escape from an admissible set of states. On the contrary, we need a guaranty that X_0 is controllable. Therefore we instead compute an under-approximation of this set.

5.1.1 Under-approximation of the controllable set

Let q be a discrete mode verifying: $0 \in D_q$. We want to compute an under-approximation of the controllable set inside the cell D_q $(\in \mathbb{R}^n)$ when $u \in \mathbb{U}_m = Conv\{s_1, \dots, s_k\}$. In the next, $X[0, s_i](.)$ denotes the trajectory according to $u = s_i$ that goes through 0 for $i = 1, \dots, k$; then $X_{q,i}$ denotes the first intersection of this trajectory with one of the guards of D_q:

$$X_{q,i} = X[0, s_i](T_i)$$

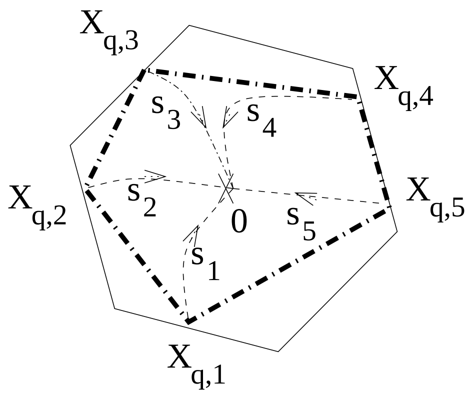


Figure 4: Under-approximation in state q of the controllable set

where: $\forall i = 1, \ldots, k,\ T_i = \sup\{t < 0 / \ X[0, s_i](t) \in \partial D_q\}$ as is shown on figure 4.

By construction of our hybrid approximation, the dynamic inside D_q is linear both in the state and the control. So, as shown in [26, Proposition 3], the controllable domain inside D_q is a convex subset of the state space:

Proposition 6. *[26] Every point of $Conv(X_{q,1}, \ldots, X_{q,k})$ is controllable. $Conv(X_{q,1}, \ldots, X_{q,k})$ is an under-approximation of the controllable set in time $T = \max_{1 \le i \le k} T_i$*

From each guard of D_q, we pursue the under-approximation, the same way. The difference is just that the reverse starting point is not 0 any more, but the extremal points of the intersection between the guard ∂D_q and the current under-approximation. The algorithm is illustrated on figure 5.

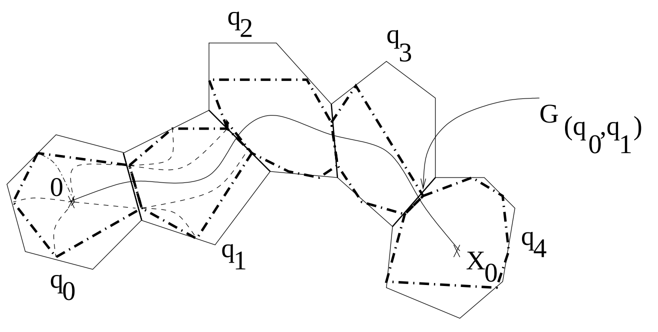


Figure 5: Construction of an under-approximation of the controllable set in a given path $q = (q_0, q_1, q_2, q_3, q_4)$ of discrete modes - X_0 is controllable

Example 2 (In dimensions $n = 3$ and $m = 2$).

$$\dot{X}(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} X(t) + \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} u(t)$$

in the domain $D = Conv(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix})$
with $\mathbb{U}_2 = Conv(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ -1 \end{bmatrix})$.

In this example, our under-approximation algorithm is performed on the state domain D, which is a simplex in $\mathbb{R}^3$ belonging to the mesh build in §4.2. Figure 6 shows the approximation of the controllable set to reach a given target: it coincides with the intersection of the domain D and the controllable set of the considered system without any state constraints.

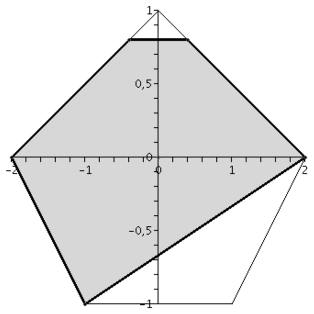


Figure 6: Under-Approximation in dimensions $n = 3$, $m = 2$

Example 3 (Some runtimes). *The under-approximation algorithm, implemented in Maple, is performed on a simplex in $\mathbb{R}^n$ for $u(t) \in \mathbb{U}_m = Conv(s_1, \ldots, s_k)$ and $A = I_n$.*

n	2	3	4	5	6
cpu (s)	0.1816	0.2048	0.2356	0.3057	0.3870

Table 3: Under-Approximation timings

Algorithm 8 UnderApproximation

```
Require: H, $q = (i)_{i=1..r}$ a sequence of discrete modes.
Ensure: an under-approximation $\Lambda$ of the controllable domain.
1: Initial under-approximation: $\Lambda := \emptyset$;
2: Target: $K := \{0\}$;
3: i:=1;
   {In each cell $D_i$, computation of the set of the reachable points by time reversal from the vertices of $K$}
4: while $i \leq r$ and $K \neq \emptyset$ do
5:    $\Sigma := \emptyset$;
      {For each vertex of $K$}
6:    for all time step j (from 1 to card(K)) do
7:       j-th vertex of $K$: M:=K[j];
         {Computation of the intersection time between trajectories from $M$ according to $u = s_p$ and the boundary of the current cell $D_i$}
8:       $T_p := sup\{t < 0; X_{q_i}[M, s_p](t) \in \partial D_{q_i}\}, p = 1..k$ where: $\mathbb{U}_m = Conv(s_1, \ldots, s_k)$.
         {$\Sigma$ is the set of the so-computed intersection points}
9:       $\Sigma := \{X_{q_i}[M, s_p](T_p); p = 1..k\} \cup \Sigma$;
10:   end for
11:   $\Lambda := \Lambda \cup Conv(\Sigma)$;
12:   $K := \begin{cases} Vertices(Conv(\Sigma) \cap G_{i,i+1}) & \text{if } i < r \\ \emptyset & \text{otherwise.} \end{cases}$
13:   i:=i+1;
14: end while
15: Return $\Lambda$.
```

5.1.2 Controllability of the initial point

Let X_0 be a given initial point in $\mathbb{R}^n$. We now want to define the controllability of X_0. To perform our under-approximation algorithm, we need to first compute the sequence of adjacent boxes $D_{q_0}, \ldots, D_{q_r}$ such that: $X_0 \in D_{q_r}$ and $0 \in D_{q_0}$. This problem leads us to introduce the notion of solution of our hybrid problem $\mathcal{P}_{\mathcal{H}}$:

Definition 5. *$(X(.), u(.))$ is a solution of the hybrid control problem ($\mathcal{P}_{\mathcal{H}}$) (i.e. X_0 controllable) if there exists a finite execution $\chi = (\tau, q, X)$ satisfying:*

i. $(q(\tau_0), X(\tau_0)) = (q_0, X_0)$ *such that:* $X_0 \in D_{q_0}$.

ii. $\forall i, X(.)$ *is continuously differentiable,* $q(t) = q_i$ *and* $X(t) \in D_{q_i}$ *over* $]\tau_i, \tau_{i+1}[$ $(\tau_i < \tau_{i+1})$

iii. $\forall i = 1, \ldots, r, X(\tau_i) \in G_{(q_{i-1}, q_i)}$

iv. $(q(\tau_{r+1}), X(\tau_{r+1})) = (q_r, 0)$

where: $\tau = (\tau_i)_{0 \leq i \leq r+1}$ *(*$\tau_0 = 0$*) and* $q = (q_i)_{0 \leq i \leq r}$.

From this definition, the difficulty is to determine the optimal sequence of modes. Some directions to solve this problem include numerical pre-simulations as done in [4] or a variable change $ds = l(X(t), u(t))dt$ to come down to a time optimal control problem. From now on, we then consider the following assumption:

Hypothesis 2. *let $q = (q_i)_{i=1\dots r}$ be a given admissible sequence of discrete modes i.e. there exists (τ, X), such that $\chi = (\tau, q, X)$ is a finite execution of the hybrid automaton $\mathcal{H}$ that steers the initial point X_0 to the target* 0.

5.2 Local optimal solutions in each cell

We thus are given a sequence of discrete modes $q = (q_i)_{1\dots r}$ in our hybrid automaton (H) as defined in hypothesis 2. In this section, we want to analyze the dynamic behavior of our hybrid approximation in one given mode q_i. According to previous construction, in this cell we define a constrained affine optimal control problem $(\mathcal{P}_{q_i})$:

Minimize the cost function $J(X_0, u(.)) = \int\limits_0^{t_f} l(X(t), u(t))dt$ with respect to the control $u(.)$ under the dynamic:

$$\begin{cases} \dot{X}(t) = A_{q_i}X(t) + B_{q_i}u(t) + c_{q_i} \\ X(0) = X_0 \end{cases} \quad X(t_f) \in G_{(q_i, q_{i+1})}$$

and the constraints: $\forall t \in [0, t_f], (X(t), u(t)) \in \Delta_{q_i}$, where the final time t_f is unspecified.

We then are typically in the context of affine optimal control problems. Methods and algorithms have been developed in [26] to solve that kind of problems via their Hamiltonian formulations: the principle is to come down to an optimization problem in the control. Indeed we first introduce he Hamiltonian function:

$$H_{q_i}(X, u, \lambda) = l(X, u) + \lambda^T(A_{q_i}X + B_{q_i}u + c_{q_i})$$

The system is subjected to both state and mixed affine inequality constraints (induced by D_{q_i} and U_{q_i}, see §4.3): $C_{q_i}(X(t)) \leq 0$ and $(M_{q_i}(X(t), u(t)) \leq 0)$. Optimal control under state inequality constraints is a hard and subtle problem. However we can show that, in the context of our hybrid approximation, as soon as a inequality constraint is broken, the trajectory switches from the present mode to the next one. In consequence, we are inside a new mode without any constraints at the new initial time. We therefore directly apply the algorithms described in part I.

The constrained affine optimal control problem $(\mathcal{P}_{q_i})$ is actually a linear program, so that optimal solutions (X^*, u^*) occur on the edges of the cell Δ_{q_i} (see e.g. figure 7). We can thus deduce that the mixed inequality constraints are satisfied as long as the state constraints are not broken. Moreover once a state constraint is broken, the guard of the mode q_i is reached and the system switches into the next mode q_{i+1} (and a new local optimal control problem $(\mathcal{P}_{q_{i+1}})$).

In each cell of our hybrid automaton (H), we have come down to a local affine optimal control problem solved with part I.

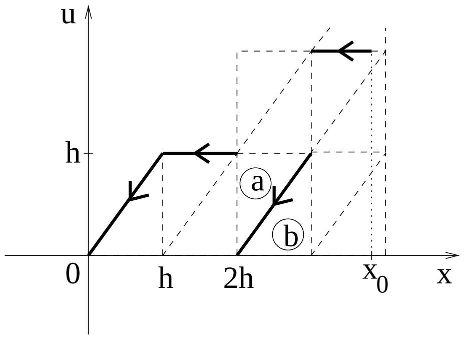


Figure 7: Example of admissible trajectory in $\mathbb{R} \times \mathbb{U}_1$

5.3 Choice of cell for a given phase location

In this section, we propose an algorithm for solving the hybrid optimal control problem $(\mathcal{P}_{\mathcal{H}})$. Our algorithm is based on a piecewise resolution of our hybrid optimal control problem. However to a given initial point in the state space correspond several possible cells Δ_q and as many possible local optimal control problems. Therefore to build our algorithm, we first have to define a method for choosing the optimal cell Δ_q.

Let $X_0 \in \mathbb{R}^n$ be a given initial point of the state space. The main step for solving the local optimal control problem is to consider a "column" of cells. More precisely, the algorithm has three steps:

1. Computation of the simplex $D_0 \in \mathcal{D}$ such that: $X_0 \in D_0$ as explained in §4.2.

2. Test if $X_0 \in D_0$ is controllable thanks to the UnderApproximation algorithm (in the next steps, X_0 is assumed controllable).

3. Consider the set $Q(X_0) = \{q; p_{\mathbb{R}^n}(\Delta_q) = D_0\}$. It is actually a "column" of cells Δ_q, whose projection in the state space is D_0 (see property (P_2) in §4.2). Just note that $\mathbb{U}_m$ is assumed bounded, so that $card(Q(X_0)) < +\infty$.

The idea is to solve the local optimal control problem $(\mathcal{P}_q)$ in each Δ_q as presented in previous section 5.2 ; and then to compute the local optimal trajectory (X_q^*, u_q^*) and the related value function $V_q(X_0)$. After that, we perform a finite discrete optimization over these cells, i.e. find the mode q_0 such that:

$$V_{q_0} = \min_{q \in Q(X_0)} V_q(X_0)$$

However the last optimization step can sometimes be avoided. Indeed, as explained in section 5.2, any optimal trajectory $(X(.), u(.))$ evolves along the edges of our mesh Δ of $\mathbb{R}^n \times \mathbb{U}_m$. Let us take the example of the figure 7 and consider the trajectory between the cells a and b. We assume that we have already computed the optimal control in each cell so that:

- in cell a, the optimal control is constant, equal to h.
- in cell b, the optimal control depends on the state and: $u^* = x - 3h$

As shown on figure 7, the optimal trajectory in cell b evolves along the boundary between the two cells a and b. In consequence, it could belong to both cells a and b. Moreover we have seen that the optimal trajectory in cell b is not optimal in cell a. We so conclude that the cell a is a best choice than the cell b for our optimal control problem.
So the discrete optimization can be avoided by the application of that kind of analysis (see figure 8).

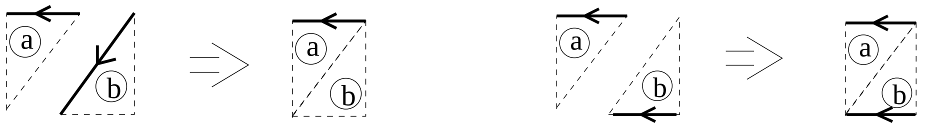


Figure 8: Example of optimal choices between two cells in dimension 2

5.4 Hybrid solver

In regard of previous results, we can now describe the HybridSolving algorithm: as expressed in hypothesis 2, we assumed a sequence of state given. The principle of our solving algorithm is to compute cells columns by cells columns a piecewise optimal solution of our hybrid problem.

Algorithm 9 HybridSolver

```
Require: $X_0$, H, $q = (q_i)_{i=1..r}$ a sequence of discrete modes s.t. $X_0 \in D_{q_0}$ and $0 \in D_{q_r}$.
Ensure: $(\tau, X, u)$, $V(X_0)$
    where $(\tau, q, X)$ is a local optimal execution of H, $V(X_0)$.
 1: if $X_0 \notin$ UnderApproximation(H,q), then
 2:     Return "$X_0$ may not be controllable".
 3: end if
 4: $\tau_0 := 0; V := 0;$
    {Piecewise Affine Resolution}
 5: for all time step i (from 1 to r) do
 6:     {Linear Approximation}:
        $(A, B, c) := (A_{q_i}, B_{q_i}, c_{q_i})$ (see eq (10))
 7:     {Output Condition}: target $:= G_{(q_i, q_{i+1})}$;
 8:     {Mixed and State Inequality Constraints}
        $U := U_{q_i}; D := D_{q_i}$;
 9:     Solve the affine problem $\mathcal{P}_{q_i} \to (X(.), u(.), t_f, V_f)$
10:     $X_0 := X(t_f); \tau_{i+1} := \tau_i + t_f; V := V + V_f$;
11: end for
12: Return $(\tau, X, u, V)$.
```

Example 4. *Let us consider the nonlinear dynamical system:*

$$\dot{x}(t) = x^2(t) + x(t)u(t) + u(t)^2 \tag{12}$$

The problem is to control the system (12) in minimum time from a given initial *point $X_0 \in \mathbb{R}$ to the origin 0 under the control constraint: $\forall t \geq 0, -1 \leq u(t) \leq 1$.*

Exact resolution

Let us consider the vector field: $f(x,u) = x^2+xu+u^2$ as a polynom of degree 2 in the variable u. Its discriminant is: $-3x^2 < 0$; we so deduce: $\forall (x,u), f(x,u) > 0$. Hence $\forall t \geq 0, \dot{x}(t) > 0$, so that $x(.)$ is increasing and the controllability conditions: $x_0 \leq 0$. We can also numerically check that the exact controllable domain of our problem is $]-\infty, 0]$.

Let us introduce the Hamiltonian: $H(X,u,\lambda) = 1+\lambda(x^2+xu+u^2)$. According to the Pontryagin minimum principle ([24]), we come down to the problem of minimizing H with respect to u under the constraints:

$$\begin{aligned} &\dot{\lambda}(t) = -2x(t)\lambda(t) - \lambda(t)u(t) \\ &H(X(t),u(t),\lambda(t) = 0 \textit{ along the optimal trajectory} \end{aligned} \tag{13}$$

Let us first assume that the optimal control takes values in $]-1,1[$. We so have to solve:

$$\frac{\partial H}{\partial u} = 0 \Leftrightarrow \lambda x + 2\lambda u = 0$$

under the condition: $\frac{\partial^2 H}{\partial u^2} > 0$. As $H \equiv 0$ along the optimal trajectory, we necessarily have: $\lambda \neq 0$ (otherwise: $H \equiv 1$), so that: $u^* = -\frac{x}{2}$ while $-2 \leq x \leq 2$ (to fulfill the control constraints). In consequence, we can now explicitely solve the system (12) and we obtain:

$$x(t) = 4x_0(4-3x_0t)^{-1} \textit{ and } \lambda(t) = -\frac{4}{3}x(t)^{-2}$$

In consequence $\lambda(t) < 0$ and the previous u^* is not minimizing H. Hence: $\forall t \geq 0, u(t) \in \{-1,1\}$.

To choose between $u^* = -1$ and $u^* = 1$, we compute the value function associated to each possible control. The comparison of these functions shows that $u^* = -1$ is the exact solution of our initial nonlinear optimal control problem and the value function is:

$$V(x_0) = -\frac{1}{9}(6 \; arctan(\frac{1}{3}(-1+2x_0)\sqrt{3}) + \pi)\sqrt{3}$$

Hybrid Approximation

Let $h = 1/N$ our discretization step.
Let us build a simplicial mesh $(D_k)_k$ of $\mathbb{R}$ (see methods in §4.2): here we have: $D_k = [kh,(k+1)h]$.

Then the control domain $[-1,1]$ is subdivise into N intervals $[u_i,u_{i+1}]$, $i = 0 \ldots N-1$, where: $u_i = -1+2ih$. To ensure that 0 would be a vertex of our triangulation (see property (P_1)), N is assumed even.

Lastly we just have to triangulate $[kh,(k+1)h] \times [u_i,u_{i+1}]$ and we can so define our global mesh Δ of $\mathbb{R} \times [-1,1]$ by:

$$\begin{aligned} \Delta^{(1)}_{k,i} &= \{(x,u); 0 \leq u-u_i \leq x-kh \leq h\} \\ \Delta^{(2)}_{k,i} &= \{(x,u); 0 \leq x-kh \leq u-u_i \leq h\} \end{aligned}$$

We now perform the hybrid approximation f_h over the cells $\Delta_{k,i}^{(j)}$, $j \in \{1,2\}$:

$$f_h(x,u) = a_{k,i}^{(j)} x + b_{k,i}^{(j)} u + c_{k,i}^{(j)}, \text{ for } (x,u) \in \Delta_{k,i}^{(j)}$$

where:
$a_{k,i}^{(1)} = 2kh + h - 1 + 2ih$
$b_{k,i}^{(1)} = -2 + 3h + kh + 4ih$
$c_{k,i}^{(1)} = -k^2h^2 + kh - 2ih^2k - 1 + 4ih - 4i^2h^2 - h^2k + 3h - 6ih^2$

$a_{k,i}^{(2)} = 2kh + 3h - 1 + 2ih$
$b_{k,i}^{(2)} = kh - 2 + 4ih + 2h$
$c_{k,i}^{(2)} = -k^2h^2 + kh - 2ih^2k - 1 + 4ih - 4i^2h^2 - 3h^2k + 2h - 4ih^2$

We have so define the hybrid automaton related to the considered optimal control problem. In the next, we will so apply the algorithm described in §5.
Let x_0 be a given initial point: $x_0 \in D_k$ with $k = E[\frac{x_0}{h}]$ and a fixed sequence of state $q = (D_j)_{j=0\ldots k}$; We then want to compute the optimal solution of the hybrid problem over the path q:
As presented in §5.2, the first step is to solve the local affine optimal control problem in each cell of Δ and we have:

$$u^* = \begin{cases} u_i & \text{when } (x,u) \in \Delta_{k,i}^{(1)} \\ x + u_i - kh & \text{when } (x,u) \in \Delta_{k,i}^{(2)} \end{cases}$$

After that, we have to find the optimal cell $\Delta_{k,i}^{(j)}$, $j \in \{1,2\}$, $i = 0,\ldots,N$ to consider. We then can notice that for all i, $\Delta_{k,i}^{(1)}$ and $\Delta_{k,i}^{(2)}$ play respectively the roles of the cells a and b of the figure 8, so that:

$$u^* = u_i \text{ when } (x,u) \in \Delta_{k,i}^{(1)} \cup \Delta_{k,i}^{(2)}$$

Recursively, we can so conclude that over the whole columns on cells $\Delta_{k,i}^{(j)}$, $j \in \{1,2\}$, $i = 0,\ldots,N$, the optimal choice is the cell $\Delta_{k,0}^{(1)}$ with $u^* = -1$. We can then compute the local value function $V_k(x_0)$.
x_0 is then re-initialized to kh in the cell D_{k-1}, and so one.

So our HybridSolving algorithm returns the optimal solution to our hybrid optimal control problem:
$u^* = -1$ *and* $V_h(x_0) = \sum_{j=0}^{k} V_j(x_j)$ *where:* $\begin{cases} x_0 \text{ given} \\ x_j = (k-j)h, \ j = 1\ldots k \end{cases}$

Figure 9 shows in dot the real value function $V(x_0)$ and two hybrid value functions for two differents discretization steps h. Our approximation converges towards the real curve.

Conclusion

In this report, we have presented a full analysis and implementation for solving both linear and complex optimal control problems.

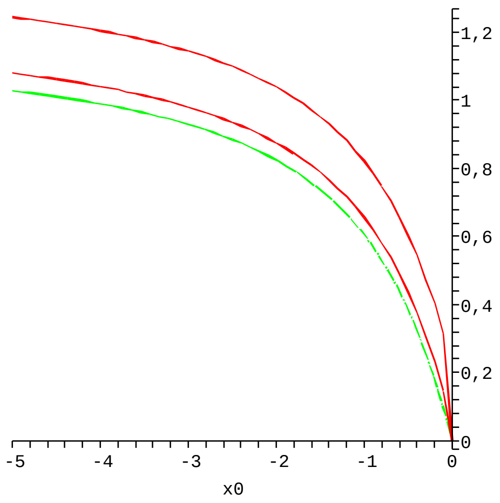


Figure 9: Hybrid approximations of the exact value function (in green dashed-dots) for: $h = .1$ and $h = 0.05$

First we have presented an algorithm for solving general linear optimal control problems: we propose an explicit method to transform any problem into a canonical one by the way of a block Kalman decomposition. We have also developed generic algorithms solving the canonical problem even when complex boundaries occur. Yet, two important new features of our algorithms are that we give a full generic implementation and most of the algorithms are symbolic. Further developments already are in progress:

- Complete the whole algorithm for a cost function nonlinear in the control. In this case, the Hamiltonian optimization problem could be solved by classical tools. Indeed, $\frac{\partial H}{\partial u} = 0$ can now be solved in the control variable.
- The UnderApproximation and solving algorithms have been performed on linear dynamical systems under the canonical form where $A_2 = 0$ (cf §2.2 and section 3). These two algorithms have to be extended for any canonical form (cf (4)). In practice, this corresponds to the appearance of a perturbation time function $t \to A_2 e^{A_3 t} Y_2(0)$ in the dynamical system. The technique does not change, but practical implementations are slightly more complex.
- The UnderApproximation could be refined and a study of the approximation error has still to be made. The idea is to consider cases where the dynamical system for $u = s_i$ admits one (or an infinite number of) equilibrium point P_i (note that 0 is an equilibrium point when $u = 0$). The under-approximation can e.g. be completed by the convex hull of trajectories from P_j that go through (or tend towards) P_j by time reversal according to $u = s_i$. Also, a rigorous proof of the convergence of our under-approximation towards the real controllable set has still to be

completed.

Then we achieved a hybrid optimal control of non-linear dynamical systems. First we propose a hybrid approximation, by way of a hybrid automaton with piecewise affine dynamics, of complex systems. Then we solve the affine optimal control problem and give an explicit and fully analytical algorithm.
This algorithm however guarantees only a local optimization. Next step will be to give a way to find a sequence of cells containing an optimal trajectory. As mentioned, several directions to solve this problem include:

- Exploring different sequences of states. This could unfortunately induce a combinatorial explosion.
- Partial numerical simulations could give some information on the localization of an optimal trajectory and thus reduce the exploration.
- Replace $l(X(t), u)$, the cost function with an admissible variable change $ds = l$, so that the problem comes down to a time optimal control problem. Finding the optimal sequence would then be to minimize the time to reach 0. To do this, one can use a time reversal, then perform an attainability test on X_0. The first time step when X_0 is reached gives an estimation of the total time and moreover the direction from which it was reached. This direction is the new goal of the affine optimal control. When a guard is reached in this direction, this gives a new point X_1, closer to zero than X_0, and from which we iterate the same loop.
- Another idea would be to perform an optimal control in the whole space with the local affine system. This would also give a possible direction. Unfortunately, convergence is then not guaranteed anymore.

Further developments are also:

- $F(X, U)$ could have a fixed point which disappears in F_h. This would augment the approximation error. A way to avoid this phenomenon is to augment the resolution. For the fixed point at 0 this could be avoided by setting 0 on a node of the mesh. At other fixed points, multi-resolution meshing could become handy.
- An exhaustive study of the approximation error, similar to [14, Theorem 10.2.2], has still to be made.

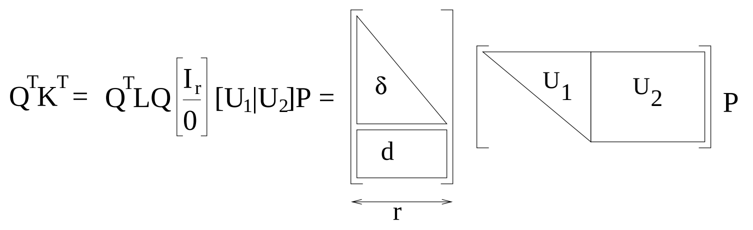
Appendix

A Proof of theorem 2

0. Facts: the LQUP factorization [16] of K^T gives:

$$K^T = LQ \left[\begin{array}{c|c} U_1 & U_2 \\ \hline 0 & 0 \end{array}\right] P = (LQ \left[\begin{array}{c} I_r \\ \hline 0 \end{array}\right])[U_1|U_2]P \tag{14}$$

Then, from [10, section 4.3], $Q^T K^T$ is the product of a permutation, a lower and an upper triangular matrix:



This shows that $G = \delta[U_1|U_2]P$. Now, from its definition, we see that the matrix K satisfies that any block column vector of AK is a linear combination of the block column vectors of K (since A^n is a linear combination of the A^i, namely that of its characteristic polynomial). Well this is also true of the permuted independent columns of K. In other words, there exist a linear combination Γ such that $GA^T = \Gamma G$ (see e.g. [15, §7.4] or [21, §3] for more details).

1. We now have to prove that $GA^T = C_1 G$. Unfortunately, C_1 might be different form Γ since G is not square. But the fact that there exist a solution is sufficient as we will see next. First, $C_1GP^T = C_1\delta[U_1|U_2]$. Then, by the definition of C_1, we obtain $C_1GP^T = GA^TP^T \left[\frac{I_r}{0}\right] [I_r|U_1^{-1}U_2]$. We replace now GA^T by ΓG to prove that the latter permutes. Then $C_1GP^T = \Gamma\delta[U_1|U_2]PP^T \left[\frac{I_r}{0}\right] [I_r|U_1^{-1}U_2]$, so that $C_1GP^T = \Gamma\delta[U_1|U_2] = \Gamma GP^T$. We thus get $C_1G = \Gamma G = GA^T$ since the permutation matrix P is invertible.

2. Then note that the inverse of $\left[\frac{G}{[0|I_{n-r}]P}\right] = T^T$ is

$$T^{-T} = P^T \left[\begin{array}{cc} U_1^{-1}\delta^{-1} & -U_1^{-1}U_2 \\ \hline 0 & I \end{array}\right]. \tag{15}$$

This enables to compute $T^{-1}AT$ or, equivalently, $T^TA^TT^{-T}$ as follows:

$$\begin{aligned} T^TA^TT^{-T} &= \left[\frac{G}{[0|I_{n-r}]P}\right] A^TT^{-T} = \left[\frac{GA^T}{[0|I_{n-r}]PA^T}\right] T^{-T} \\ &= \left[\frac{C_1G}{[0|I_{n-r}]PA^T}\right] T^{-T} \\ &= \left[\begin{array}{cc} C_1GP^T \left[\frac{U_1^{-1}\delta^{-1}}{0}\right] & C_1G\left[\frac{-U_1^{-1}U_2}{I}\right] \\ C_2 & C_3 \end{array}\right] \\ &= \left[\begin{array}{cc} C_1 & 0 \\ C_2 & C_3 \end{array}\right] \text{ as } GP^T = [\delta U_1|\delta U_2]. \end{aligned}$$

3. There only remains to prove that $(T^{-1}B)^T = [B_1|0]$: just remark that $B^TT^{-T} = [I_m|0]K^TT^{-T}$ which, from (14) and (15), gives

$$\begin{aligned} B^TT^{-T} &= [I_m|0](QQ^T)LQ\left[\frac{I_r}{0}\right][U_1|U_2]PT^{-T} \\ &= [I_m|0]Q\left[\frac{\delta}{d}\right][\delta^{-1}|0] = [I_m|0]Q\left[\begin{array}{c|c} I_r & 0 \\ \hline d\delta^{-1} & 0 \end{array}\right] \end{aligned}$$

Moreover if B is assumed full rank (equal to $m \le r$), the permutation Q has no influence on the m first rows of K^T. In this case, B_1 is just the identity.

B Proofs of the propositions

B.1 Proof of proposition 2

Let us state: $X = TY$. $(Y^*, \tilde{u}^*)$ is an optimal solution of problem 5, so that: $\tilde{V}(Y_0) = \tilde{J}(Y_0, \tilde{u}^*)$. Let us prove that the control $\Psi\tilde{u}^*$ is optimal in the problem (3):

$$\begin{array}{rcl} J(X_0, \Psi\tilde{u}^*) & = & \int_0^{+\infty} l(X(t), \Psi\tilde{u}^*(t))dt \\ & = & \int_0^{+\infty} \tilde{l}(Y(t), \tilde{u}^*(t))dt \\ & = & \tilde{J}(Y_0, \tilde{u}^*) = \tilde{V}(Y_0) \end{array}$$

Moreover, $\Psi\tilde{u}^*$ is an admissible control for (3), so, by definition of the value function: $J(X_0, \Psi\tilde{u}^*) \geq V(X_0)$. But, if (3) possesses some solutions from X_0, then there exists a control u^* such that: $V(X_0) = J(X_0, u^*)$. So, according to table 1:

$$\begin{array}{rcl} V(X_0) & = & \int_0^{+\infty} l(X(t), u^*(t))dt = \int_0^{+\infty} \tilde{l}(Y(t), \Phi u^*(t))dt \\ & = & \tilde{J}(Y_0, \Phi u^*) \geq \tilde{V}(Y_0) \end{array}$$

The only possibility is then $J(X_0, \Psi\tilde{u}^*) = \tilde{V}(Y_0) = V(X_0)$

B.2 Proof of proposition 3

We take some controllable points $(Y_i)_{1\leq i\leq k} \in C^k$ and another point $Y_0 \in Conv(Y_i; 1 \leq i \leq k)$; by convexity: $\exists(\alpha_i)_{1\leq i\leq k} \in [0,1]^k, \sum_{i=1}^k \alpha_i = 1$ and $Y_0 = \sum_{i=1}^k \alpha_i Y_i$. We want to prove that Y_0 is controllable. By definition, each Y_i satisfies:
$\exists T_i \geq 0, \exists u_i : [0,T] \rightarrow U_m, Y_i = \int_0^{T_i} e^{-A\tau}Bu_i(\tau)d\tau$. Consequently, $Y_0 = \sum_{i=1}^k \alpha_i \int_0^{T_i} e^{-A\tau}Bu_i(\tau)d\tau$. We then fix T to the maximal time T_i and define a new control as follows:
$\hat{u}_i(\tau) = \left\{ \begin{array}{lll} u_i(\tau) & if & \tau \leq T_i \\ 0 & if & T_i < \tau \leq T \end{array} \right.$ and $u(\tau) = \sum_{i=1}^k \alpha_i \hat{u}_i(\tau)$.
Clearly $u(.)$ is an admissible control function (i.e. $\forall\tau \geq 0, u(\tau) \in \mathbb{U}_m$) and we conclude: $Y_0 = \int_0^T e^{-A\tau}Bu(\tau)d\tau$, hence: $Y_0 \in C$.

B.3 Proof of proposition 4

We consider an ij-singular trajectory at a time $t \in [t_0, t_1]$. Let us show that: $u(t) \in [s_i, s_j]$. As we know, the control is defined on a polyhedral convex set, so that: $\exists(\alpha_k(t))_{k=1\ldots m} \in [0,1]^m, \sum_{k=1}^m \alpha_k(t) = 1$ and $u(t) = \sum_{k=1}^m \alpha_k(t)s_k$. So we come down equivalently to minimize the Hamiltonian H with respect to $\alpha = (\alpha_k(t))_{k=1\ldots m, k\neq i}$. H is redefined by: $H = l(X) + \lambda^T f(X) + \lambda^T g(X)s_i + \sum_{k=1,k\neq i}^m \alpha_k(t)\lambda^T g(X)(s_k - s_i)$ Moreover according to the definition 4 of an ij-singular trajectory, we have: $\forall k \neq i,j,\ \lambda^T g(X)(s_k - s_i) > 0$. So necessarily, we must have: $\forall k \neq i,j,\ \alpha_k(t) = 0$ i.e.: $u(t) \in [s_i, s_j]$. The control of an I-singular trajectory is shown likewise to belong to $Conv(s_k)$.

B.4 Proof of proposition 5

By assumptions, the trajectory $Y(.)$ is regular over $[T-\epsilon, T]$. So we have: $\exists k \in \{1,\ldots,m\}, \forall t \in [T-\epsilon, T],\ u(t) = s_k$. Moreover $u(.)$ is the optimal control, so that (cf definition 1): $\forall t \in [T-\epsilon, T],\ S_{k,j}(t) < 0$ and consequently: $\forall\epsilon' \in]0,\epsilon[,\ \forall t \in [T-\epsilon', T],\ S_{k,j}(t) < 0$.

Moreover $\lambda(.)$ is a continuous time function, so does $S_{k,j}$; therefore we have: $S_{k,j}(T) \leq 0$ when $\epsilon' \to +\infty$. Moreover the singular trajectory's definition 4 involves: $S_{i,j}(T) = 0$ and $S_{j,k}(T) < 0$. Hence we necessarily have: $k \in \{i, j\}$.

References

[1] E. Asarin, T. Dang, and A. Girard. Reachability of non-linear systems using conservative approximations. In *Proceedings of the 2003 Hybrid Systems: Computation and Control*, pages 20–35. Springer, April 2003.

[2] R. Bellman. *Dynamic Programming.* Princeton University Press, 1957.

[3] M. Bergounioux. *Optimisation et contrôle des systèmes linéaires.* Dunod, 2001.

[4] J.F. Bonnans and S. Maurin. An implementation of the shooting algorithm for solving optimal control problems. *Technical Report RT-0240, INRIA*, 2000.

[5] A.E. Bryson and Y. Ho. *Applied Optimal Control.* Hemisphere, 1975.

[6] D.F. Delchamps. *State Space and Input-Output Linear Systems.* Springer-Verlag, 1988.

[7] J. Della Dora, A. Maignan, M. Mirica-Ruse, and S. Yovine. Hybrid computation. In *Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation.* Bernard Mourrain editor, ACM Press, July 2001.

[8] J.-G. Dumas and A. Rondepierre. Modeling the electrical activity of a neuron by a continuous and piecewise affine hybrid system. In *Proceedings of the 2003 Hybrid Systems: Computation and Control*, pages 156–171. Springer, April 2003.

[9] Jean-Guillaume Dumas, Thierry Gautier, and Clément Pernet. Finite field linear algebra subroutines. In Teo Mora, editor, *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, Lille, France*, pages 63–74. ACM Press, New York, July 2002.

[10] Jean-Guillaume Dumas, Pascal Giorgi, and Clément Pernet. FFPACK: Finite Field Linear Algebra Package. In Jaime Gutierrez, editor, *Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation, Santander, Spain.* ACM Press, New York, July 2004.

[11] R. Fierro, A. K. Das, V.Kumar, and J. P. Ostrowski. Hybrid control of formations of robots. In *IEEE Proceedings of the International Conference on Robotics and Automation*, 2001.

[12] H. Freudenthal. Simplizialzerlegungen von beschraeukter flachkeit. *Annals of Math. in Science and Engin.*, 43:580–582, 1942.

[13] A. Girard. Approximate solutions of ordinary differential equations using piecewise linear vector fields. In *Proceedings of the 2002 Computer Algebra in Scientific Computing.* Springer Verlag, September 2002.

[14] A. Girard. *Analyse Algorithmique des Systèmes hybrides.* PhD thesis, Institut National Polytechnique, Grenoble, 2004.

[15] Gene H. Golub and Charles F. Van Loan. *Matrix computations.* Johns Hopkins Studies in the Mathematical Sciences. The Johns Hopkins University Press, Baltimore, MD, USA, third edition, 1996.

[16] O.H. Ibarra, S. Moran, and R. Hui. A generalization of the fast lup matrix decomposition algorithm and applications. *Journal of Algorithms*, 3:45–56, 1982.

[17] C.D. Johnson and J.E. Gibson. Singular solutions in problems of optimal control. *IEEE Transactions on Automatic Control*, 8:4–15, 1963.

[18] R.E. Kalman. Canonical structure of linear dynamical systems. In *Proceedings of the National Academy of Sciences*, pages 596–600, 1961.

[19] R.E. Kalman. Mathematical description of linear dynamical systems. *Siam Journal on Control*, 1:152–292, 1963.

[20] H.J. Kelley, R. Kopp, and H. G. Moyer. *Singular Extremals*, pages 63–101. Academic Press, 1967.

[21] Clément Pernet. Calcul du polynôme caractéristique sur des corps finis. Master's thesis, Université Joseph Fourier, jun 2003.

[22] H.J. Pesch. A practical guide to the solutions of real-life optimal control problems. *Parametric Optimization. Control Cybernet*, 23:7–60, 1994.

[23] E.R. Pinch. *Optimal Control and the Calculus of Variations.* Oxford University Press, 1993.

[24] L. Pontryagin, V. Boltiansky, R. Gamkrelidze, and E. Michtchenko. *Théorie mathématique des processus optimaux.* Editions de Moscou, 1974.

[25] H.M. Robbins. A generalized legendre-clebsch condition for the singular cases of optimal control. *IBM Journal of Research and Development*, 11(4):361–372, 1967.

[26] Aude Rondepierre and Jean-Guillaume Dumas. Algorithms for hybrid optimal control. parti: Symbolic/numeric control of affine dynamical systems. submitted.

[27] B. D. Saunders. Black box methods for least squares problems. In *Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation.* Bernard Mourrain editor, ACM Press, July 2001.

[28] M.I. Zelikin and V.F. Borisov. Optimal chattering feedback control. *Journal of Mathematical Sciences*, 114(3):1227–1344, 2003.

[29] J. Zhang, K.H. Johansson, J. Lygeros, and S. Sastry. Zeno hybrid systems. *International Journal of Robust and Nonlinear Control*, 11:435–451, 2001.