Numerical Stabilisation of Non-linear Systems: Exact Theory and Approximate Numerical Implementation

M. Alamir and N. Marchand
Laboratoire d'Automatique de Grenoble, Saint Martin d'Hères, France

In this paper, a theoretical background is presented for the stabilisation of non-linear systems. A numerical implementation is then proposed. The class of systems concerned with the proposed practical approach is quite large and contains all flat systems as a particular subset. The stabilising strategy is based on path generation strategy and avoids the integration of the differential system. The numerical implementation extensively uses the interpolation on a function basis. Two examples of systems known to be hard to stabilise are given to illustrate the proposed algorithm.

Keywords: Non-linear systems; Numerical implementation; Path generation; Stabilisation

1 Introduction

The stabilisation of non-linear systems remains quite a difficult issue as long as no special structure is assumed to hold for the non-linear system to be stabilised.

One of the most general control strategies that enables, at least conceptually, stabilisation of a very wide class of non-linear systems is the receding horizon control [1,6,9,10]. Indeed, roughly speaking, the only assumptions needed for this strategy to apply are the stabilisability of the system and some classical regularity assumptions.

Unfortunately, for general non-linear systems, the computation of the receding horizon control law implies the minimisation at each sampling time of a global non-convex open-loop cost function with final state constraint. Only the 'first part' of the corresponding open-loop optimal control is applied and the whole procedure is reiterated at the next sampling time to obtain a state feedback.

The use of repeated open-loop computations with eventually a revising feedback [14], in order to formulate a state feedback that underlines the receding horizon strategy, has also been used in the framework of differentially flat systems [2,3]. Indeed, in this context, the open-loop trajectories of the states are parametrised by the trajectory of what is called the flat output. This enables generation of open-loop trajectories satisfying stable asymptotic behaviour while avoiding integration of system equations. The state feedback is then constructed in order to control the dynamics of the tracking error with respect to this a priori defined open-loop trajectory.

Steering a system from an initial state to a final one after some time T has already been studied in the case of chained form systems [11–13]. Some methods using sinusoidal, piecewise constant or polynomial inputs can be found in the literature. A certain class of functions to describe the generated trajectory (with smooth properties) has also been used for robot path generation [5]. In this paper, we combine the receding horizon approach with path generation, which avoids integration of the

Correspondence and offprint requests to: M. Alamir, Laboratoire d'Automatique de Grenoble UMR 5528, CNRS-INPG-UJF, ENSIEG-BP 46, 38402 Saint Martin d'Hères Cedex, France.
Email: Mazen.Alamir@inpg.fr

Received 17 January 1997; Accepted in revised form 23 December 1998
Recommemded by D. Aeyels and A. Isidori
system dynamics in order to construct a stabilising feedback.

It is worth noting that, in the classical receding horizon scheme, even when one is concerned only with stabilisation — no tracking is needed — the attractivity of the origin is ensured by the minimisation procedure. Indeed, the corresponding Lyapunov function is just the optimal cost; therefore, the repeated optimisation ensures its decrease. Here, we propose a slightly different strategy in order to avoid unnecessary heavy computations related to optimisation when only stabilisation is addressed.

The open-loop problem that is systematically solved is that of finding open-loop control that steers the system to the origin. The decreasing property associated with cost minimisation in classical receding horizon control is replaced by a particular initialisation of the procedure that searches for such admissible open-loop control. This particular initialisation uses the result of the last past time search to feed the search procedure with an a priori ‘decreasing’ candidate open-loop control.

The paper is organised as follows. The aim of the paper is clearly defined in Section 2. Some definitions and notations used in the sequel are then summarised in Section 3. Section 4 presents the exact theoretical framework with a formal proof of the attractivity of the origin for the closed-loop system. A basic version is first demonstrated and then modified in order to take into account eventual disturbances. The problem of asymptotic stabilisation using the proposed scheme is also briefly addressed. Finally, Section 5 proposes an effective numerical implementation, with several examples. While the theoretical proof remains conceptually valid for any system, under the reachability condition, the numerical implementation needs some structural assumptions on the non-linear system to be satisfied. At the end of the paper, three examples are given to illustrate the efficiency of the proposed approach.

2. Problem Formulation

Consider the general non-linear system given by

$$f \left( x, \frac{dx}{dt}, u, \frac{du}{dt}, \frac{d^{n-1}u}{dt^{n-1}} \right) = 0 \quad \text{(1)}$$

where for all vector functions \(v(t) \in \mathbb{R}^q\)

$$\frac{dv_1}{dt}, \frac{dv_2}{dt}, \ldots, \frac{dv_q}{dt} = \mathbb{R}^q \quad \text{(2)}$$

\(x \in X \subset \mathbb{R}^n\) is the state of the system, \(u \in \mathbb{R}^m\) is the control input and \(X\) is a region of interest.

$$f: \mathbb{R}^{2n+mn} \rightarrow \mathbb{R}^q$$

such that \(f(0) = 0\). We shall denote by \(x(t; t_0, x_0, u(.))\) the solution at instant \(t\) of (1) when it exists under the control \(u(.)\) with initial conditions \(x_0\) at instant \(t_0\). Furthermore, we suppose that the dynamic system (1) satisfies the following assumption:

**Assumption 1.** There exists a finite time \(t_{\text{min}}\), such that, for all initial state \(x_0 \in X\), there exists an open-loop control strategy \(u(.)\) defined over \([0, t_{\text{min}}]\) that steers the state of the system to 0, namely \(x(t_{\text{min}}; 0, x_0, v(.)) = 0\).

Let \(T > 0\) be a sampling time. The aim of this work is to find an implementable state feedback \(u(t+nT) = K(x(nT), t)\) that globally stabilises the dynamic system (1) at 0. Throughout the paper, we shall take a fixed \(T \geq t_{\text{min}}\).

3. Definitions and Notations

- For the ease of the reader, we shall denote \((B)^{4} := \{f(.): A \rightarrow B\}\)
- **Assumption 1** enables us to define for all states \(x_0\) and all \(t_f \geq t_{\text{min}}\): \(A(x_0, t_f)\) as the set of all control strategies that steers the state to 0 at \(t_f\):

\[A(x_0, t_f) := \{u(.) \in (\mathbb{R}^m)^{|0,T|} \]

s.t. \(x(t_f; 0, x_0, u) = 0\}\]

In the foregoing, a fixed value \(t_f\) will be considered, hence \(A(x_0, t_f)\) will simply be denoted by \(A(x_0)\).
- Let us define a time translation function \(S: (\mathbb{R}^q)^{|0,T|} \times [0, t_f] \rightarrow (\mathbb{R}^q)^{|0,T|}\) by

\[S(u, D)(t) = \begin{cases} u(t + D) & \text{for } t \in [0, t_f - D] \\ 0 & \text{for } t \in [t_f - D, t_f] \end{cases}\]

- For all time functions \(z(t)\), we shall denote by

\[F(z(t)) = 0 \quad \text{(4)}\]

a general relation between \(z(t)\) and a finite number
of its derivatives at instant \( t \). For example, according to this notation, Eq. (1) can be formally written as follows:
\[
\dot{f}(x(t), a(t)) = 0
\]
- For all \( \eta = (\eta_1, \eta_2, ..., \eta_q) \in A_1 \times A_2 \times ... \times A_q \), we shall note \( \eta = \pi(\eta) \). Furthermore, for \( J = \{i_1, ..., i_k\} \) where \( i_1, ..., i_k \in [1, q] \), we denote
\[
\pi_x(\eta) = \begin{pmatrix} \eta_{i_1} \\ \vdots \\ \eta_{i_k} \end{pmatrix} \tag{5}
\]

4. Theoretical Background

In this section, the theoretical background that underlies the proposed approach is presented. First, a basic version is studied in Section 4.1 while in Section 4.2 a slightly improved version (with respect to unmeasured perturbations) is suggested. Strictly speaking, the proposed approach leads to a state feedback that renders the origin attractive which does not guarantee a Lyapunov-like boundness of the transient behaviour. Section 4.3 briefly discusses this feature.

4.1. Basic Version

Suppose in a quite abstract manner that there is a set \( P \) of parameters that enables us to describe the elements of \( \mathcal{A} := \bigcup_{x_0 \in X} \mathcal{A}(x_0) \) (set of admissible open-loop controls when \( x_0 \) spans \( X \)) such that the following mappings exist:
\[
U : X \times P \to X \times (\mathbb{R}^m)^{0,l}
\tag{6}
\]
\[
R : X \times \mathcal{A} \to X \times P
\tag{7}
\]
and are such that
\[
U \circ R = \text{Id}_{X \times \mathcal{A}}
\tag{8}
\]
Namely, \( U \) computes the control strategy over \([0,t_f]\) from the knowledge of an initial state and a control parametrisation, while \( R \) computes a parametrisation of a given control strategy that steers the state from a given initial state to 0. Note also that \( U \) and \( R \) keep unchanged the initial condition \((x_0)\) passed as first argument.

Suppose in addition that one has a systematic procedure that computes the parametrisation of an admissible control strategy in \( \mathcal{A}(x_0) \), namely
\[
Q : X \times P \to X \times P \text{ such that }
U(Q(x,p)) \in \{x\} \times \mathcal{A}(x) \quad \forall x(p) \in X \times P \tag{9}
\]
In other words, \( Q \) is a systematic procedure that finds an open-loop control strategy that steers the state from \( x \) to 0 during time \( t_f \) (the definition of \( \mathcal{A}(x) \)). The variable \( p \) is an initial guess for the systematic procedure \( Q \) which may be an iterative routine. Note that \( Q \) also keeps unchanged the first argument \((x)\).

We shall suppose that \( Q \) satisfies the following assumption:

**Assumption 2.** For all \( x \in X \) and all \( p \in P \)
\[
\{U(x,p) \in \{x\} \times \mathcal{A}(x)\}
\Rightarrow \{Q(x,p) = (x,p)\}
\tag{10}
\]
In other words, if one initialises the procedure \( Q \) with a ‘good’ guess, \( Q \) terminates and gives this initial guess as a solution.

This property of function \( Q \) is the key feature that ensures the attractiveness of the origin in the proposed scheme.

We have the following theorem:

**Theorem 4.1.** Let \( T > 0 \) be a sampling time. In the absence of disturbances and under Assumptions 1 and 2, the feedback law
\[
u(nT + t) = K_n(t) \quad 0 \leq t < T
\tag{11}
\]
where
\[
K_n := \pi(U \circ Q(x(nT), p_n)) \in (\mathbb{R}^m)^{0,l}
\tag{12}
\]
\[
p_n := \pi(R(x(nT), S(K_{n-1}, T))) \in P, n \geq 1
\tag{13}
\]
\( p_0 \) is arbitrarily chosen in \( P \)
\tag{14}
exists and steers the system to the origin after a finite time \( t_f \).

The crucial rule of Assumption 2 should be emphasised since it ensures the attractivity of the origin in the proposed scheme. Thanks to it, if one initialises the procedure \( Q \) with a good parametrisation \( p \) (in the sense that the state will vanish after time \( t_f \)), this strategy will be kept and hence the state of the system will reach the origin after time \( t_f \). This would no longer be true if one changes strategies at each sampling time.

**Proof.** The existence of the feedback law is obvious to the extent that, from Assumption 1, \( \mathcal{A}(x) \neq \emptyset \). Given Assumption 2, (9) and (12) imply
\[ K_n \in \mathcal{A}(x(nT)) \quad \forall n \geq 0 \]  

(15)

Furthermore, it is clear from the definition of \( \mathcal{S} \) that the assumption of a free perturbation evolution \( \text{[and } f(0) = 0] \) give

\[ [K \in \mathcal{A}(\xi)] \Rightarrow [S(K,T) \in \mathcal{A}(x(0;\xi;K))] \]  

(16)

Hence, for all \( n \geq 1, \)

\[ K_{n-1} \in \mathcal{A}(x((n-1)T)) \]  

(15) \Rightarrow \quad (17)

\[ \Rightarrow [S(K_{n-1},T) \in \mathcal{A}(x(nT))] \]  

(10)

(13) yields

\[ (x(nT),p_n) = R(x(nT),S(K_{n-1},T)) \]  

(18)

Applying \( U \) to both sides of \( (18) \) and using \( (8) \) and \( (17) \) gives

\[ U(x(nT),p_n) = (x(nT),S(K_{n-1},T)) \]  

\[ \in (x(nT)) \times \mathcal{A}(x(nT)) \]  

(19)

This implies, according to \( (10) \)

\[ Q(x(nT),p_n) = (x(nT),p_n) \]  

(20)

(19), (20) and (12) imply

\[ K_n = S(K_{n-1},T) \]  

(21)

We have by definition of the feedback law

\[ u(t + nT) = K_n(t) \]  

(21)

\[ = K_{n-1}(t + T) \text{ according to (21)} \]  

\[ = K_{n-2}(t + 2T) = \ldots = K_0(t + nT) \]  

(22)

and therefore \( u(t) = K_0(t) \). Furthermore, \( K_0 \) is an admissible open-loop control that steers the state to the origin during time \( t \) since it belongs to \( \mathcal{A}(x(0)) \).

(22) proves that the actual input resulting from the feedback is exactly \( K_0 \), which ends the proof. \( \square \)

4.2. Disturbances Handling

The feedback law (11)–(14) may behave dangerously in the presence of disturbances. Indeed, according to this feedback, during the time interval \([nT,(n+1)T]\) we apply an open-loop control depending on the state \( x(nT) \). Therefore, if a ‘bad’ perturbation arises at a time \( t^* \) in the interval \([nT,(n+1)T]\) such that the corresponding autonomous evolution has an escape time lower than \([(n+1)T - t^*]\) the state may diverge to \( \infty \) even before the application of control at the next sampling period.

Note that the above problem is a common problem that arises each time a continuous feedback law is implemented numerically. Indeed, a numerically applied feedback is always piecewise constant; therefore, during sampling time intervals one always has an open-loop control.

The reason why we shall handle this problem explicitly while it is commonly neglected lies in the fact that the computation of the feedback law (11)–(14) may need a time which cannot by indefinitely reduced.

We shall define a ‘bad perturbation’ indicator as follows. Let \( p_0 := \pi_0(Q(x_0,p^*)) \) be an admissible control parametrisation that corresponds to a perturbation-free closed-loop evolution according to Theorem 4.1. Define

\[ M(x_0,p_0) := \sup_{0 \leq t \leq t_0} \| x(t;0;x_0, U(x_0,p_0)) \]  

(23)

\[ M(x_0,p_0) \] is clearly the minimum radius of a ball in \( \mathbb{R}^n \) that contains the perturbation-free closed-loop trajectory starting from \( x_0 \) and applying the control corresponding to the initial admissible parametrisation \( p_0 \).

The definition of \( M(x_0,p_0) \) enables us to define for all \( t \in [0,T] \) a ‘bad perturbation’ indicator by

\[ e(t) := \| x(t) \| - \lambda M(x_0,p_0) \]  

(24)

\( \lambda > 1 \) is given a fixed security margin.

Note that, according to the definition of \( M(x(nT),p_n) \), during a perturbation-free evolution we have

\[ e(t) < (1 - \lambda)M(x(nT),p_n) < (1 - \lambda)\| x(T) \| \]  

(25)

A ‘bad perturbation’ is therefore defined with respect to a fixed security margin \( \lambda > 1 \). It causes the state trajectory to leave the ball \( B(0,\lambda M(x(nT),p_n)) \) \( \mathbb{R}^n \) at an instant \( \tilde{t} \in [nT,(n+1)T] \); at the same instant, \( e \) changes from negative to positive.

When this happens, the ‘escape’ assumption is considered to be sufficiently plausible to re-examine the strategy defined at instant \( nT \) to be applied over the interval \([\tilde{t},(n+1)T]\). Therefore, instant \( \tilde{t} \in [nT,(n+1)T] \) becomes a new ‘decision instant’. By decision instant \( \tilde{t} \), we mean an instant where a new admissible open-loop strategy is recomputed starting from \( x(\tilde{t}) \) to be applied during the interval \([\tilde{t},(n+1)T]\).

A consequence of this strategy is that decision instants are no more necessarily of the form \( nT \). That is why we define the following time variable ‘last decision instant’ \( D \) as follows:

\[ D := \max \left\{ \frac{1}{T} \left[ \frac{\tilde{t}}{T} \right] \right\} \]  

(26)
where \( E\left(\frac{t}{T}\right) \) is the integer part of \( \frac{t}{T} \) and \( \bar{t} \) is the last instant the indicator \( e(t) \) changed from negative to positive with \( e(t) \) redefined for all \( t \in [0, T] \) as follows:

\[
e(D + t) := \| x(D + t) \| - \lambda M(x(D), p(D)) \tag{27}
\]

where \( p(D) \) is the last computed control parametrisation, \( \bar{t} \) being initialised to 0. Note that the initial value of \( D \) is clearly equal to 0 and equations (26) and (27) enable one to correctly define \( D \). Note also that we always have \( t - D \leq T \).

**Remark 4.1.** (23) can be defined separately for each component of the state \( x \), which enables one to detect ‘bad’ perturbation more accurately.

Now we have all that is needed to formulate the stabilising feedback law:

**Theorem 4.2.** Let \( D \) be defined by (26)–(27) and \( D^* \) denote the value of \( D \) before its last change (remember that \( D \) changes in a discontinuous manner). The feedback law defined by

\[
u(D + t) = K_D(t); \quad 0 \leq t < T \tag{28}
\]

where

\[K_D := \pi_2(U \circ Q(x(D), p(D))) \tag{29}\]

\[p(D) := \begin{cases} 
\pi_2(R(x(D), S(K_{D^*}, D - D^*))) & \text{if } D > 0 \\
\text{Arbitrarily chosen } \in \mathcal{P} & \text{if } D = 0 
\end{cases} \tag{30}
\]

globally asymptotically stabilises (1).

**Proof.** Note first that if there are no perturbations, the feedback law (28)–(30) is exactly the same as that of Theorem 4.1 since under this assumption one has \( D = nT \).

Therefore, the proof is straightforward according to the above discussion and the result of Theorem 4.1. Indeed, by introducing the indicator \( e \) and the non-uniform decision instants through \( D \), we avoid the perturbation-caused ‘explosion’ of the state. The ‘steering’ property of the feedback law (Theorem 4.1) enables a conclusion.

\[\square\]

It is worth noting that at \( t = 0 \) one has \( D = D^* = 0 \) and the initial parametrisation for the procedure is arbitrarily chosen in \( \mathcal{P} \) so that (29)–(30) properly define \( K_D \).

4.3. About Lyapunov Stability

The feedback strategy proposed in this section can be summarised as follows. In the absence of disturbances, the closed-loop control is chosen so that the closed-loop path is identical to the open-loop path. This simple choice is sufficient to ensure the attractiveness of the origin. With the present assumptions, nothing prevents the open-loop controls from steering the state to the origin with large excursions.

In [7], the so-called ‘admissible open-loop path generators’ are used to design an asymptotically stabilising state feedback (in the Lyapunov sense). Besides some regularity assumptions, the above path generators are assumed to satisfy a somehow transitivity assumption (see Definition 2.1 of [7]) that can be closely related to Assumption 2 mentioned above. Under these conditions, the state trajectory over an infinite time interval can be used to construct a Lyapunov function. Furthermore, in the absence of disturbances, open-loop and closed-loop trajectories coincide again.

5. Numerical Implementation

In the preceding section, we have presented an abstract theoretical framework that leads to a stabilising feedback for general non-linear systems. The aim of the present section is to propose a corresponding practical numerical implementation that holds for a particular class — although wide — of non-linear systems including systems that are not state feedback linearisable (see examples below). For these systems, the proposed method leads to a systematic and parameterised path-planning algorithm.

The class of systems showing particular structural properties and concerned by the implementation is presented in the first subsection. Examples of such systems are then given, emphasising the wideness of the proposed class. The principle of the method is then described.

5.1. Definition of Systems under Consideration

Let us consider the system defined by (1). Note that it can be written in an abstract manner \( f(\xi(t)) = 0 \). The following property characterises the class of systems concerned by the implementation proposed in the remainder of the paper:

**Definition 5.1.** The set of equations
\[ f(\mathbf{z}(t)) = 0 \] (31)
is said to be in normal form if there exist

a subdivision of \( \mathbf{z} = \begin{pmatrix} \mathbf{z}_f \\ \mathbf{z}_D \end{pmatrix} \) with,
\( \mathbf{z}_f \in \mathbb{R}^{n_f}, \mathbf{z}_D \in \mathbb{R}^{n_D} \) and \( n_f \neq 0 \) such that the constraint (31) is compatible with a completely free a priori choice of \( \mathbf{z}_f(.) \) as a time function (as far as the initial constraints are respected);
\( n_D \) relations \( F_i(\mathbf{z}_f, \mathbf{z}_D(t)) = 0 \) such that the set of constraints (31) is equivalent to
\[ F_i(\mathbf{z}_f(t), \mathbf{z}_{D1}(t), \mathbf{z}_{D2}(t), \ldots, \mathbf{z}_{Dn}(t)) = 0 \] (32)

1 \leq i \leq n_D

Such a normal form shows particular triangular properties of the system and consequently allows one to compute successively \( \mathbf{z}_{D1}, \mathbf{z}_{D2}, \ldots, \mathbf{z}_{Dn} \). Furthermore, as the foregoing examples will underline, particular forms of (32) can simplify the calculation (e.g. when (32) is linear in \( \mathbf{z}_D \), rendering the search of an open-loop trajectory easier).

5.2. Examples

In this section, four illustrative examples are exposed. The first two examples concern classes of systems that are well known in the non-linear control literature, namely the flat systems and the chained form systems. The last two examples are typically non-flat systems [2,3]. In Marchand and Alamir [8], the above strategy has been applied to rigid spacecraft in failure mode. The robustness with respect to parameter uncertainties has been rather successfully tested although the problem is known to be hard to solve even in an uncertainty-free context.

5.2.1. Differentially Flat Systems

All differentially flat systems [2,3] can be transformed into the normal form in the sense of Definition 5.1. Indeed, when using our notations, the system \( f(\mathbf{x}(t), \mathbf{u}(t)) \) is differentially flat if one can find \( y = h(\mathbf{x}, \mathbf{u}) \) such that there exist two functions \( G \) and \( H \) satisfying
\[ \dot{x} = G(y) \; ; \; \mathbf{u} = H(y) \]

Now, if we denote
\[ \mathbf{z} := \begin{pmatrix} y \\ x \\ u \end{pmatrix} \; ; \; \tilde{f}(\mathbf{z}) = \begin{pmatrix} y - h(\mathbf{x}, \mathbf{u}) \\ x - G(y) \\ u - H(y) \end{pmatrix} \]
then it is straightforward that \( \tilde{f}(\mathbf{z}) = 0 \) is in normal form with the following choices:
\[ \mathbf{z}_f := y \; ; \; \mathbf{z}_D := \begin{pmatrix} x \\ u \end{pmatrix} \]

Furthermore, the relations \( F_i = 0 \) become trivial algebraic equalities of the form (33) rather than time-varying differential equations:
\[ F_i(\mathbf{z}_f, \mathbf{z}_D) = G(\mathbf{z}_f) - \mathbf{z}_D = 0 \] (33)

5.2.2. Chained Form Systems

Given a general chained form system [13] with \( m + 1 \) inputs, \( m.(m+1) \) chains and \( m+1 \) generators:
\[ x_j^0 = v_j \; 0 \leq j \leq m \]
\[ x_j^i = x_j^{i-1}v_{i} \; 1 \leq i \leq m; \; 0 \leq j, i \leq m; \; j - i \]
by choosing \( z_r := (v_0, \ldots, v_m) \), the system (34) is already in the normal form in the sense of definition 5.1, and we have
\[ \mathbf{z}_D = (x_0^0, \ldots, x_m^0, x_0^{1}, \ldots, x_{m-1}^{1}, \ldots, x_0^m, \ldots, x_{m-1}^m) \]
\[ (x_j^0)_{0 \leq i < j \leq m}, \ldots, (x_j^m)_{0 \leq i < j \leq m}, (x_j^0)_{0 \leq i < j \leq m} \]

It is then obvious that the relations \( F_i \) can be deduced from (34).

5.2.3. Inverted Pendulum with Horizontal Action

Consider the equations of motion of the inverted pendulum:
\[ x_1 = x_2 \]
\[ \dot{x}_2 = L(1 - a \cos^2 x_1) \]
\[ \left( -aL \sin x_1 \cos x_2 \right) + g \sin x_1 - \frac{a}{m} \cos x_1 u \]
\[ x_3 = x \]
\[ x_4 = \frac{a}{1 - a \cos^2 x_1} \]
\[ \left( aLx_3 \sin x_1 - ag \right) \sin x_1 \cos x_1 + \frac{a}{m} u \]
That we shall write for convenience as follows:
\[
\begin{align*}
    x_1 &= x_2, \\
    x_2 &= \Phi_2(x_1, x_2) + \Psi_2(x_1)u, \\
    x_3 &= x_4, \\
    x_4 &= \Phi_4(x_1, x_2) + \Psi_4(x_1)u
\end{align*}
\]
(35)

In these equations \( x := [\theta, \dot{\theta}, r, \dot{r}]^T \) and \( u \) is the horizontal force applied on the car. It is well known that the inverted pendulum with the only horizontal force as control action is not a flat system.

System (35) is already in the normal form with, for example
\[
\begin{align*}
    z_f := x_1; \quad z_D := (x_2 u x_4 x_3)^T \\
    F_1(z_f, z_{D1}) := z_f - z_{D1} \\
    F_2(z_f, z_{D1}, z_{D2}) := z_{D2} \\
    F_3(z_f, z_{D1}, z_{D2}, z_{D3}) := z_{D3} - z_{D4}
\end{align*}
\]
(36)

5.2.4. The Ball and Beam

This system, studied elsewhere [3,4], is also known to be a non-flat system. After a trivial static feedback, the system can be given by
\[
\begin{align*}
    x_1 &= x_2, \\
    x_2 &= -Bg \sin(x_3) + Bx_2^2x_1, \\
    x_3 &= x_4, \\
    x_4 &= w
\end{align*}
\]
(40)

where \( x := [r, \theta, \dot{\theta}]^T \), \( r \) and \( \theta \) being respectively the position of the ball on the beam and the inclination of the beam from horizontal. Equations (40) are already in normal form with
\[
\begin{align*}
    z_f := w; \quad z_D := (x_4 x_3 x_1 x_2)^T
\end{align*}
\]
and
\[
\begin{align*}
    F_1(z_f, z_{D1}) := z_{D1} - z_f \\
    F_2(z_f, z_{D1}, z_{D2}) := z_{D2} - z_{D1} \\
    F_3(z_f, z_{D1}, z_{D2}, z_{D3}) := z_{D3} - Bg \sin(z_{D2}) - Bx_2^2z_{D3}
\end{align*}
\]
(41)

5.3. Principle of the Implementation

The numerical implementation is based on the extensive use of a functional basis defined on the interval \([0,1]\). In this paper, a Chebyshev polynomial basis is considered:
\[
\begin{align*}
    T_0(t) &:= 1, \\
    T_1(t) &:= 2t - 1, \\
    T_n(t) &:= 2T_{n-1}(t) - T_{n-2}(t)
\end{align*}
\]
(48)

We shall also denote by \( T(t,q,N) \) the \( \mathbb{R}^q \times N_q \) matrix defined by
\[
\begin{align*}
    T(t,1,N) &:= [T_1(t), \ldots, T_N(t)] \quad (49)
\end{align*}
\]

for \( n \geq 2 \)

\( T(t,q,N) \) clearly enables one to write the approximation of a time function \( v(t) \) on \( \mathbb{R}^q \) in a compact manner as
\[
v(t) = [T(t,q,N)] a \quad a \in \mathbb{R}^{N_q}
\]

Let us finally define for all \( i \in \mathbb{N} \) and all \( m \in \mathbb{N} \), \( T^m(t) \) to be the \( m \)th derivative of \( T(t) \) and \( T^{-m}(t) \) to be the \( m \)th primitive of \( T(t) \) that vanishes at 0. We then naturally define the corresponding \( (q \times N_q) \) matrices \( T^m(t,q,N) \) and \( T^{-m}(t,q,N) \) for all \( q \geq 1 \) according to (49).

Recall that, to the extent that system (1) can be set in the previous defined normal form, for all \( i \in [1,n_2] \), we have the triangular system of differential equations:
\[
F_i(z_f(t), z_{D1}(t), \ldots, z_{D_{i-1}}(t), z_{D_i}(t)) = 0 \quad (50)
\]

with \( z_{D_i} = \left( \frac{dz_{D_i}}{dr}, \frac{d^2z_{D_i}}{dr^2}, \ldots \right) \)

(51)

Let \( a \) be the vector of parameters that defines the evolution of \( z_f \), i.e.:
\[
z_f(t) = T(t,q,N) a \quad a \in \mathbb{R}^{N_q}
\]

(52)
The inverted pendulum problem

Computed command (Initial state: [5° 0 0 0])

Behaviour of the states (Initial state: [5° 0 0 0])

Initial state 0 = [6.673 0 0 0]

Lambda = 2.000
Perturbation checking period = 0.01 s

Fig. 1. Inverted pendulum with horizontal action.

The key idea to transform the $n_D$ relations (50) into an algebraic system of equations in the unknown $a$ (see Eq. (54) below), expressing the fact that $z_D(t_i) = 0$ for $i \in [1,n_D]$.

For all $i \in [1,n_D]$, let $\alpha_i$ be the coordinates of the best least-squares approximation of the solution of differential equation (50), namely:

$$z_D(t) = T(t,1,N_D)\alpha_i(a)$$

$\alpha_i$ being the solution of the following least-squares problem:

$$\alpha_i(a) = \text{Arg}\left\{\min_{\alpha \in \mathbb{R}^D} \sum_{j=1}^{N_L} \| F_j(z)(t_j), \alpha_j \| \right\}$$

Furthermore, depending on the nature of $z_D$, $\alpha_i$ may have to respect initial conditions on $z_D$. Writing the final constraint $z_D(t_f) = 0$ for $i \in [1,n_D]$ gives the following triangular system of static non-linear equations:

$$NLE(t_f,x_0,a) := T(t_f,1,N_D)\alpha_i(a) = 0, i \in [l_D+1,n_D]$$

which is the non-linear system of equations in the unknown $a$ we were looking for. $l_D$ is defined as in the following remark:

**Remark 5.1.** The search space can be reduced:

1. By using the initial and final constraints on $z_D$, linear equations can then be obtained.
2. When a set of $l_D$ equations (50) yields to equations (53) that are linear in $a$, the initial and final constraints on $z_D$ give $n_{i_D}$ linear constraints on $a$. The corresponding equation $NLE(t_f,x_0,a) = 0$ is linear and allows reduction of the search space dimension.

Let us define $n_L := n_D + n_{i_D}$. The search space dimension can then be reduced from $(n_DN_f)$ to $(n_DN_f - n_L)$ by introducing a new unknown vector $p$ of independent variables. The vector $a$ will then depend on $p$. See Appendix for further details.

**5.4. Concrete Definition of the Abstract Subroutines $U$, $R$ and $Q$**

In this section, we use the results of the numerical implementation equations in order to define precisely the abstract notions defined in Section 4, namely,
Numerical Stabilisation of Non-linear Systems

The ball and beam problem

Computed command (without perturbation)

Computed command (with perturbation)

Initial state $X_0 = [0.5]$  $t_f = 2s$  $T = 1s$  $Na = 20$

Lambda = 1.100  
Perturbation checking period = 0.05 s  
Perturbation at $T_pert = 1.25$ s of $X_pert = [1.5]$  

Behaviour of the states (without perturbation)

Behaviour of the states (with perturbation)

Fig. 2. The ball and beam problem

Finite time divergence

Computed command (without perturbation detection)

Computed command (with perturbation detection)

Initial state $X_0 = [0.5]$  $t_f = 2s$  $T = 1s$  $Na = 20$

Lambda = 1.100  
Perturbation checking period = 0.1 s  
Perturbation at $T_pert = 1.25$ s of $X_pert = [0.1 0 0]$  

Behaviour of the state (without perturbation detection)

Behaviour of the state (with perturbation detection)

Fig. 3. System with a finite escape time.
the parameter space $\mathcal{P}$, and the way that one computes for all $x_0 \in X$, $p_0 \in \mathcal{P}$ and $u_0^j \in (\mathbb{R}^m)^{N_j}$ the quantities $U(x_0,p_0)$, $R(x_0,u_0^j)$ and $Q(x_0,p_0)$. This completes the definition of the feedback law proposed in Theorem 4.2. We assume that the system under consideration satisfies the assumptions of Definition 5.1, with the $F_S$ defined by (32).

It becomes clear that the parameter set $\mathcal{P}$ is that to which belongs the vector of unknowns $p$ and therefore $\mathcal{P} := \mathbb{R}^{n_p-n_{\text{f}p}-n_{\text{p}p}}$.

Recall that when $z_\varepsilon(t) = T(t, n_p, N_j) a(p)$, the evolution of $z_{d_0}$ can be obtained by solving (54). The corresponding control $u(t)$ can then be deduced.

The functions $U$ and $R$ are simply given by

$$U(x_0,p_0) = (x_0,u^0(p_0)) \quad (55)$$

$$R(x_0,u^j(\cdot)) = (x_0,p) \quad (56)$$

with $p$ being the coordinates of the projection of $z_\varepsilon(\cdot)$ on the functional basis, when the control $u^j(\cdot)$ is applied on the system.

The path search procedure $Q$ is defined as follows:

$$Q(x_0,p_0) = \text{solution over } p \text{ of } [(NLE, t, x_0, p) = 0]_{t < \text{stabilisation}}$$

$$x_0 \quad \text{with initial guess } p_0 \quad (57)$$

6. Examples

In this section, we will present the numerical results, and some elements of the implementation. For Figs 1–3 the legend ‘Perturbation at $T_{\text{pert}} = \tau_{\text{pert}}$ of Xpert $= \delta_{X_{\text{pert}}}$’ means that an additive perturbation of value $\delta_{X_{\text{pert}}} \delta(t - \tau_{\text{pert}})$ will append on the states of the system. The ‘perturbation checking period’ is the period used to check if any perturbation occurs on the system between each computation of the command.

6.1. Inverted Pendulum

For this example, the initial and final constraints on $z_f$ give $n_{z_f} = 2$. It is clear from (36) that $z_{d_1}$ linearly depends on $a$, and therefore the initial and final constraints on $z_{d_1}$ give $n_{z_{d_1}} = 2$. Furthermore, $z_{d_2}(t_f) = 0$ can be taken out from the set of non-linear equations to be solved. The least-squares problem (53) is linear because all Eqs (37) to (39) are linear differential equations with respect to $z_{d_2}$. The path-planning problem can finally be resumed in solving only three non-linear equations in the unknown $p$ by construction of the method, an initialisation very close to a solution.

Figure 1 shows the ability of the method to solve severely non-linear problems (starting angle of 50°). The simulations were done taking $L = 0.6 m$, $m = 0.125 \text{ kg}$ and $a = 0.2$.

6.2. Ball and Beam

In this example, $z_f$ is the input of the system and therefore only the final constraint can be used to reduce the search space dimension. We have $n_{z_f} = 1$. It is clear from (41) and (42) that $z_{d_1}$ and $z_{d_2}$ linearly depend on $a$, and therefore the initial and final constraints on $z_{d_1}$ and $z_{d_2}$ give $n_{z_{d_1}} = 2$. Furthermore, $z_{d_1}(t_i) = 0$ and $z_{d_2}(t_i) = 0$ can also be taken out from the set of non-linear equations to be solved. As for the inverted pendulum problem, the least-squares problem (53) is linear for all $\alpha$. The path-planning problem can in this case be resumed in solving only two non-linear equations in the unknown $p$ with an initialisation very close to a solution.

See Fig. 2 for the result with $B = 0.5$.

6.3. Example Showing the Effect of Perturbation Detection

This system was chosen to underlie the importance of perturbation detection as explained in Section 4. The equations of the system are

$$x = x^2 + u$$

Let us choose $z_f = x$ and $z_d = u$.

This system has a finite escape time. Therefore, if a bad perturbation occurs while $T > \frac{1}{x(kT)} (k \in \mathbb{N})$, the system may diverge (see Fig. 3).

7. Conclusion

In this paper, a theoretical framework for the stabilisation of general non-linear systems is proposed. A numerical implementation that approximates the theoretical framework is then given. Mainly two elements make the method work powerfully. The first one is structurally held in the method and is the way the procedure $Q$ is initialised with the previous solution (after a time translation) as defined in Section 4. The second lies in the triangular
property of the systems, defined in Section 5, to which it is possible to apply such numerical implementation. This numerical implementation is based on an extensive use of polynomial interpolation. Finally, several examples have been given in order to prove the efficiency of the proposed method. The robustness of the method as well as the problem of real time implementation (although some points were treated) remain to be addressed. However, taking into account that only approximations of the solutions of the differential equations are used (which can be seen as system uncertainties), the stability of the studied systems let us catch sight of good robustness properties.

References

7. Marchand N, Alamir M. From open loop trajectories to stabilizing state feedback: application to a CSTR. In: Proceedings of the IFAC conference on system structure and control, Nantes, France, 1998, pp 801–806


Appendix

The aim of this section is to explain how the search space dimension can be reduced. The inverted pendulum is an example of such a situation. The initial and final constraints on $z_f$ yield to the first two lines of Eq. (58). Expressing the initial and final constraints on $z_{D_1}$ = $T(1)(t_{n_f}N_f)a$ (which depends linearly on a) we obtain the two last lines of Eq. (58). Furthermore, $z_{D_1}$ can be taken out from the non-linear equation to be solved because for any choice of a the final constraint $z_{D_1}(t_f) = 0$ will be satisfied:

$$
\begin{bmatrix}
T(0, n_f N_f) \\
T(t_f n_f N_f) \\
T(1)(0, n_f N_f) \\
T(1)(t_f n_f N_f)
\end{bmatrix} a = 
\begin{bmatrix}
z_f(0) \\
0 \\
0 \\
0
\end{bmatrix}
$$

More generally, the initial constraints on $z_f$ and eventually special on $z_{D_1}$ when the $F$s are such that $z_{D_1}$ depends linearly on a, yield to the $n_L$ following linear equations.

$$Ma = b(x_0) \quad M \in \mathbb{R}^{n_L \times n_L}$$

Equation (59) can be used to reduce the search space dimension. Indeed, by eliminating dependent variables of a, one can define the new unknown vector of independent variables $p$ by

$$a = Fp + \gamma(x_0) \quad p \in \mathbb{R}^{\tilde{n}}$$

with $n_p = N_p n_f - n_L$

with

$$F := P \begin{pmatrix} -Q^{-1}R \\ I_{n_p \times n_p} \end{pmatrix}$$

$$\gamma(x_0) := P \begin{pmatrix} Q^{-1}b(x_0) \\ 0_{n_p \times 1} \end{pmatrix}$$

where $P \in \mathbb{R}^{n_p \times n_p}$ is a reordering orthonormal transformation matrix such that

$$M, P = [Q_{n_L \times n_L}, R_{n_L \times n_p}]$$

with $Q$ full rank.

The problem can now be solved in $p$ instead of in $a$. 